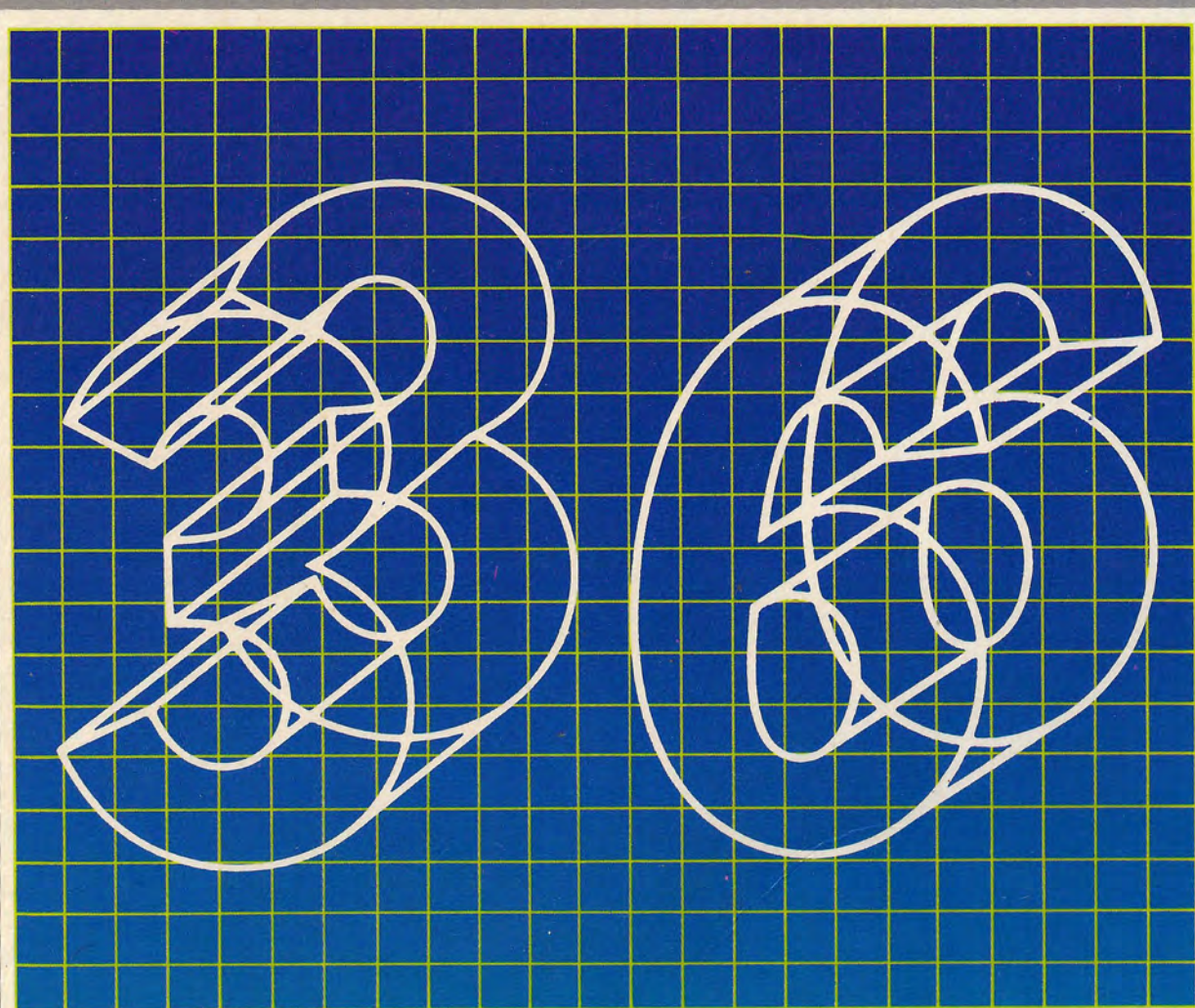


BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



BRICOLAGE INFORMATICO

MANEJO DE ELEMENTOS GRAFICOS

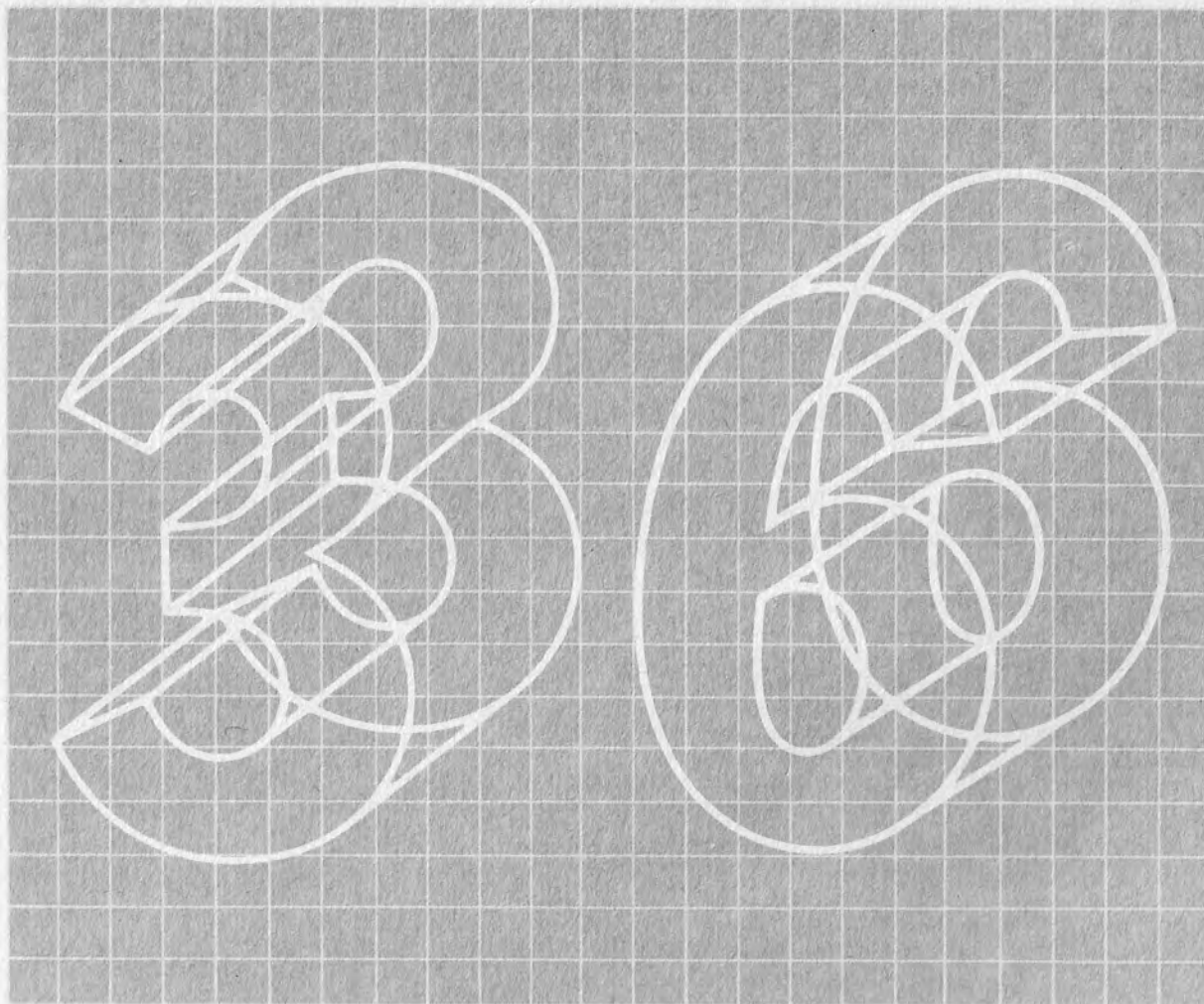
EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:

RICARDO ESPAÑOL CRESPO.

Gerente:

ANTONIO G. CUERPO.

Directora de producción:

MARIA LUISA SUAREZ PEREZ.

Director de la colección:

JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:

BRAVO-LOFISH.

Maquetación:

D. S. M.

Dibujos:

JOSE OCHOA

Tomo 36. «Experiencias y prácticas en logo», Carlos Albert, Técnico de Informática; Adoración Llena Jubelo, Técnica de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMATICA APLICADA: Francisco Javier Gavilán, Diplomado en Telecomunicación; Alejandro Marcos, Licenciado en Química, «El Taller de Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», «Vocabulario de Informática», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:

Pedro Teixeira, 8-2.º. Teléf. 255 09 99. 28020 Madrid.

Publicidad:

Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:

COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.

Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:

CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.

Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-178-9

ISBN de la obra: 84-7688-047-2

Fotocomposición:

ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:

GRAFICAS EMA. (Madrid)

© Ediciones Siglo Cultural, S. A., 1986

Depósito legal: M-43.185-1986

Printed in Spain - Impreso en España.

Suscripciones y números atrasados:

Ediciones Siglo Cultural, S. A.

Pedro Teixeira, 8-2.º. Teléf. 259 73 31. 28020 Madrid.

Diciembre, 1987

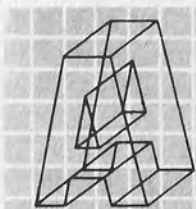
P.V.P. Canarias: 355.—Ptas.

INDICE

■ EXPERIENCIA Y PRACTICAS EN LOGO	5
■ MANEJO DE SPRITES Y ELEMENTOS GRAFICOS	14
■ TRUCOS Y RUTINAS BASICAS	22
■ EL TALLER DEL HARDWARE	31
■ APRENDER CON EL ORDENADOR	42
■ PEQUEÑA HISTORIA DE LA INFORMATICA	51
■ TEMAS MONOGRAFICOS DE VANGUARDIA	55
■ TERMINOLOGIA	59

EXPERIENCIAS Y PRACTICAS EN LOGO

■ Funciones del editor



LGUNAS versiones de Logo para los PC-compatibles permiten realizar diferentes funciones dentro del editor Logo.

Estas funciones son las de BUSQUEDA y SUSTITUCION.

■ Función de BUSQUEDA

Esta función nos permite buscar en el texto que tenemos en el editor la palabra que nosotros deseamos.

Para ello tendremos que realizar los siguientes pasos:

- Pulsar F6

En el momento que se pulse, en la pantalla aparece una ventana en donde podemos leer el mensaje:

BUSCAR?

- Escribir la palabra que deseamos buscar.

- BUSCAR AV

- Pulsar RETURN, si queremos que la búsqueda comience.

En el momento que se pulse RETURN, se comenzará a leer el texto que tenemos en el editor y cuando se encuentre la palabra a buscar, el cursor se situará detrás de ella. Podemos entonces cambiarla, borrarla, etc.

- Pulsar CTRL-F6 si queremos que continúe la búsqueda.

Si deseamos que continúe buscando la palabra indicada, pulsaremos simultáneamente las teclas CTRL y F6 y seguirá buscando a partir de la posición del cursor.

- Pulsar ESC si deseamos anular la función de búsqueda.

En cualquier momento podremos salir de la ventana de búsqueda con sólo pulsar la tecla ESC.

■ Función de SUSTITUCION

Esta función nos permite buscar una palabra que tengamos en el texto que hay en el editor, y sustituirla por la que deseamos.

Para ello realizaremos los siguientes pasos:

- Pulsar F7

En el momento que pulsamos F7, en la pantalla aparece una ventana en donde podemos leer:

BUSCAR?

- Escribir la palabra que deseamos buscar para sustituirla.

BUSCAR? AV

Una vez que hayamos escrito la palabra, pulsamos RETURN y en la ventana aparece:

BUSCAR? AV

CAMBIAR?

- Escribir la palabra por la que queremos que se sustituya la palabra a buscar.

Podemos editar procedimientos estando dentro del editor.

EXPERIENCIAS Y PRACTICAS EN LOGO

BUSCAR? AV
CAMBIAR? RE

Una vez que hayamos escrito la palabra, pulsamos RETURN y en la ventana aparece:

BUSCAR? AV
CAMBIAR? RE
OPCIONES?

— Escribir la opción que deseamos.
Las opciones que tenemos son:

— Si escribimos una T, la sustitución se hará automáticamente en todo el texto una vez que se pulse RETURN.

Comenzará a leer el texto y cada vez que encuentre la palabra AV la cambiará automáticamente por RE y continuará buscando

— Si escribimos una D, la sustitución se hará de la siguiente forma:

Comenzará a leer el texto que hay en el editor y cada vez que se encuentre la palabra a buscar, se detendrá la lectura y el cursor se situará junto a la palabra. En la pantalla aparecerá el mensaje:

CAMBIAR (S/N)

Tendremos que pulsar la tecla S si deseamos que la palabra se cambie, y pulsar N si no deseamos que se cambie.

Una vez contestada la pregunta y realizada la operación correspondiente, tenemos que pulsar las teclas CTRL y F6 si queremos que continúe la función de sustitución.

— Pulsar ESC si deseamos anular la función de sustitución.

En cualquier momento podremos salir de la ventana de sustitución con sólo pulsar la tecla ESC.

Otras dos funciones que nos facilitan estas versiones de Logo, son las de editar un procedimiento una vez que estamos dentro del editor y cargar un fichero, también estando dentro del editor.

■ Editar un procedimiento dentro del editor

Siempre que queremos editar uno o varios procedimientos determinados, utilizamos la primitiva EDITA (ED).

Otra posibilidad que tenemos es la de editar uno o varios procedimientos una vez que estamos dentro del editor y no tener que abandonarlo para hacerlo.

Para realizar esta función tendremos que:

— Pulsar F5.

Una vez pulsada, aparecerá una ventana en la pantalla con el mensaje:

EDITAR?

— Escribir el nombre del procedimiento que deseamos que se edite.

EDITAR? P1

Una vez que se pulse RETURN, el procedimiento especificado aparecerá en el editor a continuación del último procedimiento que haya en el editor.

Si deseamos editar más de un procedimiento a la vez, tenemos que especificar uno a continuación del otro los nombres correspondientes.

EDITAR? P1 P2 P2

— Pulsar ESC si deseamos anular la función de edición.

En cualquier momento podremos salir de la ventana de editar con sólo pulsar la tecla ESC.

■ Cargar un fichero Logo desde el editor

Una vez que nos encontramos dentro del editor, podemos acceder a la unidad de lectura y cargar directamente en él un fichero determinado.

Para ello realizamos los siguientes pasos:

— Pulsar F9

En el momento que se pulse F9, en la pantalla aparecerá una ventana con el mensaje:

TRAER?

— Escribir el nombre del fichero que se desea recuperar.

TRAER? CASA.LOG

Observa que el nombre del fichero tiene que ir seguido de la extensión .LOG para que pueda ser cargado en el editor.

Si el fichero que se especifica no se en-

Un fichero puede ser cargado desde el editor.

cuentra en el disco, en la ventana aparecerá el mensaje:

```
TRAER? CASA.LOG
ARCHIVO CASA.LOG NO EXISTE
```

Si el fichero es encontrado, todos los procedimientos que lo componen aparecerán en el editor, pero no quedarán definidos en la memoria del ordenador. Para que sean definidos, tendremos que abandonar el editor pulsando F1.

Si se desea cargar más de un fichero en el editor tenemos que repetir esta operación para cada uno de ellos, ya que si indicamos más de un fichero a traer:

```
TRAER? CASA.LOG COCHE.LOG
recibiremos el mensaje:
NO SE COMO HACER CON CO-
CHE.LOG
```

— Pulse ESC si deseamos anular la función de carga.

En cualquier momento podremos salir de la ventana de cargar con sólo pulsar la tecla ESC.

Una primitiva relacionada con esta última función es:

```
? TRAE "nombre de fichero
```

Los procedimientos que forman el fichero que se especifica serán cargados en el editor, pero éstos no serán definidos en el área de trabajo de la memoria. Para que queden definidos tendremos que abandonar el editor pulsando F1.

Si el fichero es demasiado grande o el editor se encuentra demasiado lleno para contener otro fichero, recibimos el mensaje:

```
EDITOR LLENO.
```

En este caso, también tenemos que indicar un solo fichero, ya que si indicamos más de uno:

```
? TRAE "CASA "MARCO
recibiremos el mensaje:
NO SE QUE DEBO HACER CON MAR-
CO
```

■ Otras primitivas

— DEFINIDO? "nombre

Devuelve CIERTO si nombre es el nombre de un procedimiento o de una primitiva, si no, devuelve FALSO.

```
? PARA "A1
> ES "LOGO
> FIN
? ES DEFINIDO? "A1
CIERTO
```

El procedimiento A1 está definido.

```
? ES DEFINIDO? "A2
FALSO
```

(Hemos supuesto que el procedimiento A2 no está definido.)

```
ES DEFINIDO? "RELLENA
CIERTO
```

La palabra RELLENA es una primitiva; por tanto, DEFINIDO? devuelve CIERTO.

El siguiente procedimiento devuelve diferentes mensajes dependiendo de la palabra que indiquemos como dato de entrada al ejecutarlo.

Si la palabra especificada no es el nombre de un procedimiento ni el de una primitiva, en la pantalla aparecerá el mensaje:

```
PALABRA NO DEFINIDA
```

Si la palabra es el nombre de una primitiva, en la pantalla aparecerá el mensaje:

```
PALABRA ES PRIMITIVA.
```

Si la palabra es el nombre de un procedimiento, el mensaje será:

```
PALABRA ES NOMBRE DE PROCEDI-
MIENTO
```

```
? PARA P :A
> SI NO DEFINIDO? :A [ES [PALABRA
NO DEFINIDA] ALTO]
> SI PRIMITIVA? :A [ES [PALABRA ES
PRIMITIVA]]
[ES [PALABRA ES NOMBRE DE PRO-
CEDIMIENTO]]
> FIN
```

Si cargas un fichero desde el editor no olvides que para que los procedimientos que lo forman queden definidos, tenemos que salir del editor pulsando F1.

EXPERIENCIAS Y PRACTICAS EN LOGO

? P "P

PALABRA ES NOMBRE DE PROCEDIMIENTO.

La palabra P es el nombre del procedimiento que acabamos de definir.

? P "AV

PALABRA ES PRIMITIVA

La palabra AV es primitiva, es la abreviatura de AVANZA.

? P "AXC

PALABRA NO DEFINIDA

? P "123

PALABRA NO DEFINIDA

? P "GIRADERECHA

PALABRA ES PRIMITIVA

NOTA: La versión LOGO del SPEC-TRUM dispone de la primitiva:

ESTA.DEFINIDO? "nombre

Devuelve CIERTO si nombre está definido como procedimiento.

Veamos otra primitiva.

— MUESTRA objeto

Presenta en la pantalla el objeto especificado y salta a la línea siguiente.

Si objeto es una lista, ésta aparecerá entre corchetes.

Observa con el siguiente ejemplo la diferencia entre esta primitiva y la primitiva ESCRIBE.

? ESCRIBE [PAPELES GRANDES]

PAPELES GRANDES

La lista especificada aparece en la pantalla sin los corchetes.

? MUESTRA [PAPELES GRANDES]

[PAPELES GRANDES]

La lista especificada aparece en la pantalla con los corchetes.

Esta primitiva muestra la naturaleza del objeto que se especifica.

? MUESTRA "AVION

AVION

? MUESTRA [TROZOS DE CRISTALES
[MUY PEQUEÑOS]]

[TROZOS DE CRISTALES [MUY PEQUEÑOS]]

? HAZ "A [EL SEMAFORO ESTA VERDE]

? MUESTRA :A

[EL SEMAFORO ESTA VERDE]

La primitiva MUESTRA acepta únicamente un dato.

? MUESTRA [LA] "NAVIDAD

[LA]

NO SE QUE DEBO HACER CON NAVIDAD

? HAZ "A "ANTIGUO

? MUESTRA "ES :A

ES

NO SE QUE DEBO HACER CON ANTIGUO

Cuadro resumen

— TRAE "nombre de fichero (PC-COMPATIBLES)

Carga en el editor Logo el fichero que se especifica, siendo necesario abandonar a continuación el editor para que los procedimientos que lo forman queden definidos en el área de trabajo de la memoria.

— DEFINIDO? "nombre

Devuelve CIERTO si nombre es el nombre de un procedimiento o de una primitiva, si no, devuelve FALSO.

— MUESTRA objeto

Presenta en la pantalla el objeto especificado mostrando su naturaleza y salta a la línea siguiente.

Ejercicios

1. Intenta dibujar una mano.

2. Define los procedimientos necesarios para hacer una quiniela con el número de apuestas que elija el usuario.

3. Define los procedimientos necesarios para que introduciendo el nombre de una provincia española, obtengamos el correspondiente prefijo telefónico e indique, con SI o NO, si tiene aeropuerto.

Podemos cargar un fichero desde fuera del editor sin que sus procedimientos queden definidos.

4. Intenta realizar el siguiente dibujo:

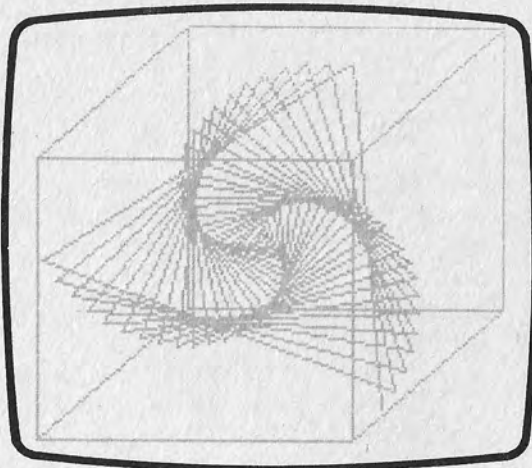


Fig. 1

5. ¿Son correctas las siguientes órdenes?:

- ? TRAE "DIBUJO BARCO
- ? PARA A
 - > TRAE "F1
 - > TRAE "F2
 - > TRAE "F3
 - > FIN
- ? PARA A1
 - > DEFINIDO? "A1
 - > FIN
- ? PARA A2
 - > HAZ "A "AHORA
 - > ES DEFINIDO? "A2 "AV
 - > FIN
- ? PARA A3
 - > HAZ "A LISTA "CASA "JARDIN
 - > MUESTRA :A
 - > FIN

■ Solución de los ejercicios

1:

Realizamos el dibujo de la mano con el siguiente procedimiento.

- ? PARA MANO
- > BP
- > SL
- > PONPOS [-40 -86]
- > GI 30
- > BL
- > PONCL 1

- > AV 130
- > PONRUMBO 55
- > REPITE 9 [AV 3 GD 10]
- > AV 45
- > PONRUMBO 12
- > AV 120
- > PONRUMBO 40
- > REPITE 14 [AV 1.5 GD 10]
- > AV 80
- > PONRUMBO 12
- > AV 100
- > PONRUMBO 45
- > REPITE 14 [AV 1.5 GD 10]
- > AV 100
- > PONRUMBO 12
- > AV 95
- > PONRUMBO 45
- > REPITE 14 [AV 1.5 GD 10]
- > AV 95
- > PONRUMBO 14
- > AV 60
- > PONRUMBO 45
- > REPITE 14 [AV 1.2 GD 10]
- > AV 80
- > GD 5
- > AV 70
- > PONRUMBO 272
- > AV 87
- > SL
- > CENTRO
- > PONCL 2
- > BL
- > RELLENA
- > FIN

Ejecuta el procedimiento tecleando:
? MANO

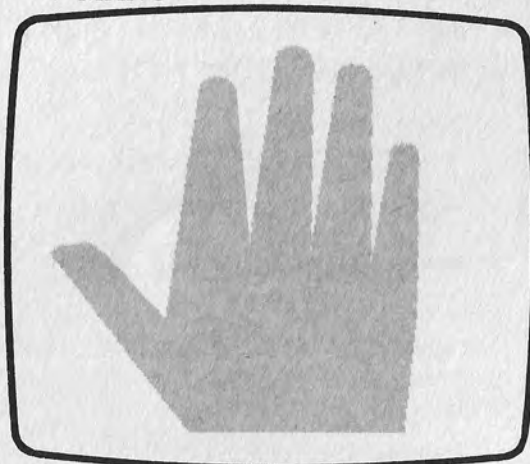


Fig. 2

Algunas versiones de Logo nos permiten buscar y sustituir palabras que estén en el editor.

EXPERIENCIAS Y PRACTICAS EN LOGO

2:

Definimos un procedimiento principal para inicializar la pantalla y para ejecutar los procedimientos que utilizaremos.

? PARA QUINIELA

```
> OT
> BP
> BT
> APUESTAS
> BP
> BT
> BOLETO :A
> FIN
```

El usuario puede elegir el número de apuestas que quiere obtener.

El procedimiento APUESTAS se encarga de ello.

El número de apuestas posibles es de 1 a 6.

? PARA APUESTAS

```
> PONCURSOR [3 7]
> ES [APUESTAS POSIBLES : 1 A 6]
> PONCURSOR [3 9]
> ES [¿NUMERO DE APUESTAS?]
> PONCURSOR [3 10]
> HAZ "A LC
> SI O :A < 1 :A > 6 [APUESTAS]
> FIN
```

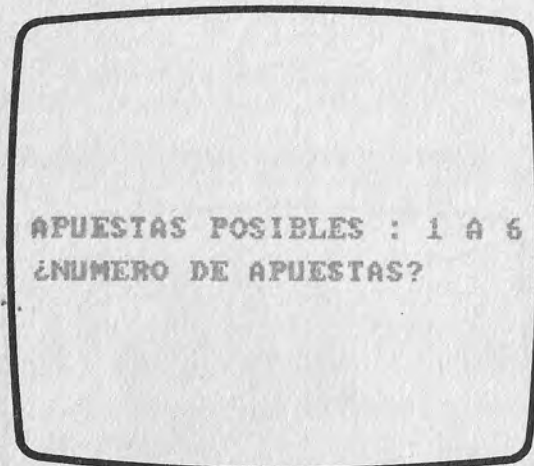


Fig. 3

El procedimiento BOLETO dibuja el boleto, dependiendo del número de apuestas que se haya elegido.

? PARA BOLETO :A

```
> SL
> HAZ "X -((:A * 32) / 2)
> HAZ "IN (40 - (:A * 4)) / 2
> PONPOS LISTA :X -64
> PONRUMBO 0
> PONCL 1
> BL
> HAZ "R :A * 32 - 8
> REPITE 2 [AV 130 GD 90 AV :R GD 90]
> PONCL 1
> REPITE 3 [GD 90 AV 8 GI 90 AV 130 RE 130]
> REPITE :A - 1 [REPITE 8 [GD 90 AV 1 PONCL 2 GI 90 AV 130 RE 130] PONCL 1 REPITE 3 [GD 90 AV 8 GI 90 AV 130 RE 130]]
> GI 90
> PONCL 1
> REPITE 3 [GD 90 AV 10 GI 90 AV :R RE :R]
> REPITE 2 [REPITE 10 [GD 90 AV 1 PONCL 2 GI 90 AV :R RE :R] PONCL 1 REPITE 4 [GD 90 AV 10 GI 90 AV :R RE :R]]
> HAZ "I 0
> REPITE :A [HAZ "W 5 JUEGO 1 :I HAZ "I :I + 1]
> FIN
```

Desde el procedimiento BOLETO se realizan tantas llamadas al procedimiento JUEGO como número de apuestas se hayan elegido.

JUEGO rellena aleatoriamente las casillas de la quiniela.

```
> PARA JUEGO :N :A
> HAZ "X :IN + :A * 4
> HAZ "M AZAR 3
> HAZ "Y :W + :N
> SI O :Y = 9 :Y = 14 [HAZ "W :W + 1]
> PONCURSOR LISTA (:X + :M) :Y
> ES "X
> SI :N = 11 [ALTO] [JUEGO :N + 1 :A]
> FIN
```

Si queremos obtener una quiniela con tres apuestas (la cuarta no nos interesa), tecleamos:

Entre las primitivas que nos informan encontramos la primitiva DEFINIDO?

? QUINIELA
y pulsamos el 3.

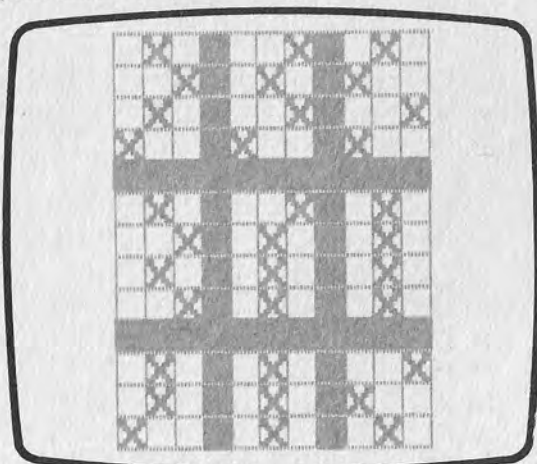


Fig. 4

Si queremos una de seis apuestas:
? QUINIELA
y pulsamos el 6.

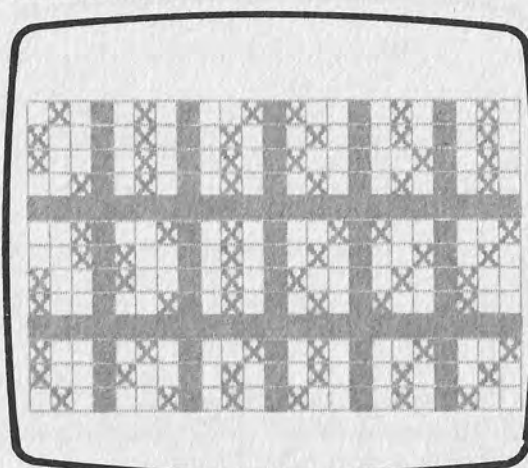


Fig. 5

En primer lugar, creamos tres listas,
una con los nombres de las provincias, otra
con los prefijos telefónicos y la última con las
provincias que tienen aeropuerto:

? PARA LISTAS

> HAZ "A [945 967 965 951
985 918 924 971 93 947
927 956 964 956 926 957
981 966 972 958 928 911
943 955 974 953 987 973
941 982 91 952 952 968
948 988 988 986 923 922 942 911 954 975 977
974

925 96 983 94 988 976 9738]
> HAZ "B [ALAVA ALBACETE
ALICANTE ALMERIA ASTURIAS
AVILA BADAJOZ BALEARES
BARCELONA BURGOS CACERES
CADIZ CASTELLON CEUTA
CIUDADREAL CORDOBA CORUÑA
CUENCA GERONA GRANADA
GRANCANARIA GUADALAJARA GUIPUZ-
COA HUELVA HUESCA
JAEN LEON LERIDA LOGROÑO
LUGO MADRID MALAGA MELILLA
MURCIA NAVARRA ORENSE
PALENCIA PONTEVEDRA SALAMANCA
TENERIFE SANTANDER SEGOVIA
SEVILLA SORIA TARRAGONA
TERUEL TOLEDO VALENCIA
VALLADOLID VIZCAYA ZAMORA
ZARAGOZA ANDORRA]
> HAZ "C [ALICANTE ALMERIA
GRANCANARIA BARCELONA
VIZCAYA CORDOBA GERONA
GIJON GRANADA BALEARES
CADIZ CORUÑA MADRID MALAGA
MURCIA ASTURIAS NAVARRA
TENERIFE GUIPUZCOA PONTEVEDRA
SANTANDER SEVILLA TARRAGONA
VALENCIA VALLADOLID ALAVA
ZARAGOZA]
> HAZ "I 1
> FIN

El procedimiento P1 es el que presenta
la pantalla en donde se indica la provincia, el
prefijo, el aeropuerto y el mensaje de petición
del nombre de una provincia:

? PARA P1

> PONCURSOR [4 5]
> ES "PROVINCIA
> PONCURSOR [16 5]
> ES "PREFIJO
> PONCURSOR [27 5]
> ES "AEROPUERTO
> SL
> PONPOS [-138 40]
> BL
> REPITE 2 [AV 40 GD 90
AV 284 GD 90]
> AV 20 GD 90 AV 284
> RE 98 GI 90 AV 20
> RE 40 GI 90 AV 96
> GD 90 AV 40

**Si se lo preguntamos, el Logo nos dice si un
procedimiento está definido o no.**

EXPERIENCIAS Y PRACTICAS EN LOGO

```
> PONCURSOR [10 1]
> ES [INTRODUCE LA PROVINCIA]
> FIN
```

P2 realiza una lectura desde el teclado de una lista en la posición adecuada. La lista introducida se le asigna la variable L:

```
? PARA P2
> PONCURSOR [3 7]
> HAZ "L LL
> ESPERA 10
> FIN
```

Una vez que se ha introducido la provincia, leemos los elementos de la lista de provincias con la ayuda del contador I. Cada elemento de esta lista lo asignamos a la variable P y lo transformamos en una lista. Cuando el elemento que se está leyendo coincide con la variable L (que es la que contiene la provincia introducida), ordenamos que se escriba el elemento que marca el contador I de la lista de prefijos:

```
? PARA P3
> SI :I > CUENTA :B [ALTO]
> HAZ "P (ELEMENTO) :I :B)
> HAZ "P LISTA :P
> SI :P = :L [PONCURSOR [16 7]
  ES ELEMENTO :I :A HAZ "I 1 P4]
  [HAZ "I :I + 1 P3]
> FIN
```

Una vez que se ha escrito el prefijo correspondiente, realizamos el mismo proceso, pero con la lista de provincias con aeropuerto:

```
? PARA P4
> SI :I > CUENTA :C [PONCURSOR
  [31 7] ES "NO ALTO]
> HAZ "A (ELEMENTO :I :C)
> HAZ "Q LISTA :Q
> SI :Q = :L [PONCURSOR [31 7]
  ES "SI] [HAZ "I :I + 1 P4]
> FIN
```

Por último, el procedimiento P5 es el que nos da opción a introducir otra provincia:

```
? PARA P5
> PONCURSOR [10 12]
> ES {[OTRA PROVINCIA? S / N]
> HAZ "T LC
```

```
> SI :T = "S [TO]
> SI :T = "N [ALTO]
> P5
> FIN
```

Una vez que tenemos todos los procedimientos necesarios para realizar el proceso, los englobamos en el procedimiento TO:

```
? PARA TO
> BP
> BT
> OT
> LISTAS
> P1
> P2
> P3
> P5
> FIN
```

Si lo ejecutamos:

```
? TO
```

Men la pantalla aparece:

PROVINCIA	PREFIJO	AEROPUERTO

INTRODUCE LA PROVINCIA

Fig. 6

Si ahora introducimos el nombre de una provincia, por ejemplo, TERUEL, la pantalla aparece de la siguiente forma:

PROVINCIA	PREFIJO	AEROPUERTO
TERUEL	974	NO

¿OTRA PROVINCIA? S / N

Fig. 7

Debemos indicar siempre qué deseamos hacer con el resultado de las primitivas que devuelven información.

Si el nombre de la provincia es compuesto, por ejemplo, CIUDAD REAL, tienes que escribirlo junto.

4:

El procedimiento A dibuja el cubo y llama al procedimiento B:

```
? PARA A
> BP
> BT
> OT
> SL
> PONPOS [-60 -60]
> BL
> REPITE 4 [AV 120 GD 90]
> AV 120 GD 45 AV 80 GD 45
> REPITE 4 [AV 120 GD 90]
> AV 120 GD 90 AV 120
> GD 45 AV 80 GD 135 AV 120
> GD 45 AV 80
> SL
> PONPOS [-60 -60]
> BL
> AV 80
> SL
> PONPOS [25 25]
> PONRUMBO -90
> BL
> B 2
> FIN
```

El procedimiento B es el que dibuja la espiral:

```
PARA B :L
> AV :L GD 122
> B :L + 2
> B
> FIN
```

Observa que este procedimiento es recursivo, por lo que puedes pararlo en el momento que desees.

Para ejecutarlo introduce:

? A

5:

? TRAE "DIBUJO "BARCO
INCORRECTO

No podemos especificar dos nombres de fichero:

```
- ? PARA A
> TRAE "F1
> TRAE "F2
> TRAE "F3
> FIN
```

Al ejecutar este procedimiento no se producirá ningún mensaje de error, pero al final sólo estará en el editor el último fichero cargado (F3)

```
- ? PARA A1
> DEFINIDO? "A1
> FIN
```

El LOGO devuelve el mensaje de error:
NO SE QUE DEBO HACER CON CIER-
TO.

No hemos indicado qué queremos hacer con lo que devuelve la primitiva DEFINIDO?

```
- ? PARA A2
> HAZ "A "AHORA
> ES DEFINIDO? "A2 "AV
> FIN
```

INCORRECTO.

El Logo muestra el mensaje de error:
NO SE QUE DEBO HACER CON AV

La primitiva DEFINIDO? acepta únicamente un dato.

```
- ? PARA A3
> HAZ "A LISTA "CASA "JARDIN
> MUESTRA :A
> FIN
```

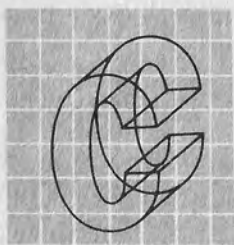
CORRECTO.

Si lo ejecutamos, en la pantalla aparecerá:
[CASA JARDIN]

El Logo dispone de tres primitivas para presentar texto en la pantalla: ESCRIBE, MUESTRA y TECLEA.

La primitiva MUESTRA presenta un objeto en la pantalla mostrando su naturaleza.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS



CONTINUANDO con la música y siguiendo con los ordenadores que nos dejamos en el tintero en el fascículo anterior, vamos a ver cómo podemos seguir haciendo de nuestros programas unas pequeñas obras de arte.

El sonido en el IBM

Aunque el IBM sólo dispone de un canal de sonido, tenemos la posibilidad de crear sonidos con dos instrucciones distintas: SOUND y PLAY. Estudiémoslas separadamente.

La instrucción SOUND

Esta instrucción nos va a permitir reproducir cualquier sonido de cualquier frecuencia durante un tiempo determinado. Necesita dos parámetros para poder funcionar: uno, es la frecuencia, y el otro, la duración del sonido. Su sintaxis es:

SOUND Frecuencia,Duración

La frecuencia ha de ir expresada en hertzios y ha de ser un número comprendido entre 37 y 32767, ambos inclusive. Cuanto mayor sea el número, mayor será la frecuencia y, por tanto, más agudo el sonido.

Si queremos saber a qué frecuencia corresponde cada nota, podemos consultar la figura 5 del tomo anterior.

Por otra parte, el tiempo lo mediremos de una forma rara, ya que cada unidad que pongamos se contará como 1,82 segundos. Esto quiere decir que una duración de 18,2 será exactamente un segundo. La duración puede ser un número entre 0 y 65535, ambos inclusive.

Una de las características más importantes de esta sentencia es que el sonido continúa hasta que aparece otra instrucción SOUND en el programa o hasta que se produce un error o un CHR\$(7).

En la figura que aparece a continuación vemos los TEMPOS musicales típicos en términos del tiempo en que se mide la duración de la sentencia SOUND.

Tempo		Pulsaciones/ Minuto	Tic-tac/ Pulsación
muy lento	Muy largo		
	Largo	40-60	27.3-18.2
	Larghetto	60-66	18.2-16.55
	Grave		
	Lento		
lento	Adagio	66-76	16.55-14.37
	Adagietto		
	Andante	76-108	14.37-10.11
moderado	Andantino		
	Moderato	108-120	10.11-9.1
	Allegro	120-168	9.1-6.5
	Vivace		
muy rápido	Veloce		
	Presto	168-208	6.5-5.25
	Prestissimo		

Fig. 1

El programa 1 nos permite hacernos una idea de cómo trabaja esta instrucción.

```

100 REM *****
110 REM * LOS SONIDOS QUE SE CREAN CON SOUND EN EL IBM *
120 REM *****
130 REM
140 CLS
150 KEY OFF
160 WIDTH 40
170 FOR D=5 TO 40 STEP 5
180   READ F
190   READ N$
200   LOCATE 12,19
210   PRINT N$;" "
220   SOUND F,N
230   SOUND 32767,5
240 NEXT D
250 REM
260 REM *** LINEAS DATA ***
270 REM
280 DATA 1048.50,"DO"
290 DATA 987.77,"SI"
300 DATA 880.00,"LA"
310 DATA 783.99,"SOL"
320 DATA 698.46,"FA"
330 DATA 659.26,"MI"
340 DATA 587.33,"RE"
350 DATA 523.25,"DO"

```

La instrucción PLAY

La instrucción PLAY se encuentra tanto en el BASIC del IBM como en el del MSX. Por ello, todo lo que aquí se diga vale para ambos.

La instrucción PLAY es la otra instrucción que nos permite crear música y sonidos en el IBM. Esta sentencia agrupa dentro de sí misma un microlenguaje específico. Su formato es el siguiente:

PLAY cadena alfanumérica

Llamamos cadena alfanumérica a una variable alfanumérica (del tipo A\$, por ejemplo) o a una serie de caracteres específicos cerrados entre comillas. Estos caracteres específicos serán letras y números que tienen un significado para el IBM, éste los interpretará y los ejecutará. Estos son:

A, B, C, D, E, F, G = Son las notas musicales según la relación que vimos en el tomo anterior y que ahora repetimos.

A = LA
B = SI
C = DO
D = RE
E = MI

F = FA
G = SOL

Después de la nota puede aparecer el signo almohadilla (#) o el signo (+) para indicar que la nota es sostenida, o bien el signo menos (-) para indicar que ésta es bemol.

PLAY "CDEFGABAGFEDC"

On = Hace que el ordenador toque en una determinada escala. Hay siete octavas numeradas desde el 0 hasta el 6, siendo la más grave la primera y la más aguda la última. Si no se especifica octava, el ordenador toma por defecto la cuarta.

PLAY "02 CDEFGAB 03 CDEFGAB 05 CDEFGAB"

<n = Nos eleva en una escala y toca la nota que le indiquemos. La nota no puede ser una letra, sino que tiene que ser un número entre 0 y 84. Para entender mejor esto, mírese la instrucción **Nn**.

PLAY "03 CDEF <80"

>n = Nos descende en una escala y toca la nota número **n**.

PLAY "03 CDEF >80"

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

Nn = Toca la nota número **n**. **n** puede ser cualquier número entero entre 0 y 84. El valor 0 indica silencio, mientras que las 84 notas restantes corresponden a las notas de las siete octavas. La nota número 1 es el DO de la octava 0.

PLAY "N1 N2 N3 N0 N3 N2 N1"

Ln = Define la duración de las notas que vienen a continuación. **n** puede tomar cualquier valor comprendido entre 1 y 64. La relación entre la notación musical y el valor de **n** es el siguiente:

L1 = Redonda
L2 = Blanca
L4 = Negra
L8 = Corchea
L16 = Semicorchea
L32 = Fusa
L64 = Semifusa

Se permiten los valores intermedios para poder especificar notas con puntillo.

PLAY "L1 CDEFG L2 CDEFG L4 CDEFG L8 CDEFG"

En el caso en que necesitemos cambiar solamente la duración de una nota, podemos poner su duración a continuación de la letra que la define.

PLAY "C1 D2 E4 F8 G16 F32"

Pn = Sirve para realizar silencios. El valor de **n** tiene el mismo significado y se usa igual que en **L**.

PLAY "CDEFG P2 GFEDC"

= El puntillo sirve para que la nota junto a la que se encuentra tenga un incremento en su duración de la mitad de su valor. Por ejemplo, si ponemos:

PLAY "L2 C."

la duración de la nota será igual a la de una blanca más una redonda.

PLAY "L2 CDEFG G.F.E.D.C."

Se pueden poner más de un punto. Por ejemplo, una blanca con dos puntos tendrá la misma longitud que una blanca más una negra más una corchea.

Tn = Le dice al ordenador con qué tiempo queremos que se ejecute la música. El valor de **n** ha de estar comprendido entre 32 y 255. Cuanto más grande sea el valor, más rápida y alegre será la música.

MF = Este subcomando sirve para que el programa no continúe hasta que se ha terminado de ejecutar la última nota. El ordenador irá leyendo nota a nota y las irá tocando una a una.

MB = Nos permite hacer que una música se esté ejecutando mientras el resto del programa sigue corriendo. El ordenador lee todas las notas de golpe y las almacena en una zona de memoria para ir las leyendo una a una.

MN = Este es el modo normal en que se encuentra el ordenador cuando lo encendemos. Cada nota sonará durante 7/8 del valor que se le asignó con **L**. Esto sirve para que se pueda apreciar la separación entre las notas.

ML = Modo ligado. Cada nota suena el total de la duración (8/8) que se especificó con **L**.

MS = Modo stacato. Cada nota suena 6/8 de la duración que se especificó con **L**. Esto sirve para dar la impresión de STACATO en la música. Dicho efecto consiste en hacer desaparecer el sonido antes de que toque hacerlo y en esperar 2/8 antes de empezar la siguiente nota.

X variable: = Nos permite ejecutar dentro de la cadena una subcadena con una música ya definida. Es algo así como incluir una subrutina dentro de la cadena de caracteres.

A\$="L4 CCC":PLAY "XA\$: L2 CDEFG XA\$:"

Hay que decir que en todos los comandos donde se utiliza un número **n**. Este puede ser sustituido por una variable numérica con sólo poner delante de ella un signo igual y después dos puntos.

A=4:B=2:C=1:PLAY "L=A: CD L=B: EF L=C: G"

El sonido en el COMMODORE

Realizar sonidos en el COMMODORE es, en cierto modo, más complejo que en otros ordenadores al no disponer de ninguna sentencia en BASIC para tal uso, pero en el fondo esto no es un problema, y si lo es, se ve recompensado por la cantidad y calidad de la música que podemos crear con este ordenador.

El COMMODORE cuenta con un chip especialmente pensado para producir sonido. Este chip no sólo nos permite tener hasta tres canales de sonido, sino que además podemos decirle que toque música mientras el programa sigue su curso.

El SID (es el nombre de este chip) cuenta con 29 registros para programar el sonido. Estos registros se encuentran situados entre las direcciones 54272 y 54300, ambas inclusive. De los 29 registros hay siete para cada canal de sonido, otros cuatro son para realizar funciones que afectan a las tres voces a la vez, y los otros cuatro sirven para realizar algunas funciones especiales. Los iremos viendo poco a poco.

Primero hay que decir que para actuar sobre cualquiera de estos registros tenemos que utilizar la sentencia POKE. Para hacerlo con más comodidad, lo haremos de la siguiente forma:

POKE 54272+Registro,Valor

Lo hacemos de esta manera porque es más fácil entender y saber qué registro estamos variando si ponemos:

POKE 54272+12,100

que si ponemos:

POKE 54282,100

aunque en la práctica sea lo mismo.

Lo primero que hay que hacer antes de programar ningún sonido es poner todos los registros a cero. Esto se consigue con un simple bucle que será necesario al principio de todos nuestros programas.

FOR I=0 TO 28:POKE 54272+I,0:NEXT I

Inmediatamente después podemos definir el volumen con el que queremos oír las melodías. El registro del volumen es el 24. En él podemos poner un número entre 0 (mínimo) y 15 (máximo).

POKE 54272+24,15

Después podemos pasar a definir la envolvente del volumen para cada una de las voces que vamos a utilizar. En la figura 2 podemos ver qué partes tiene dicha envolvente:

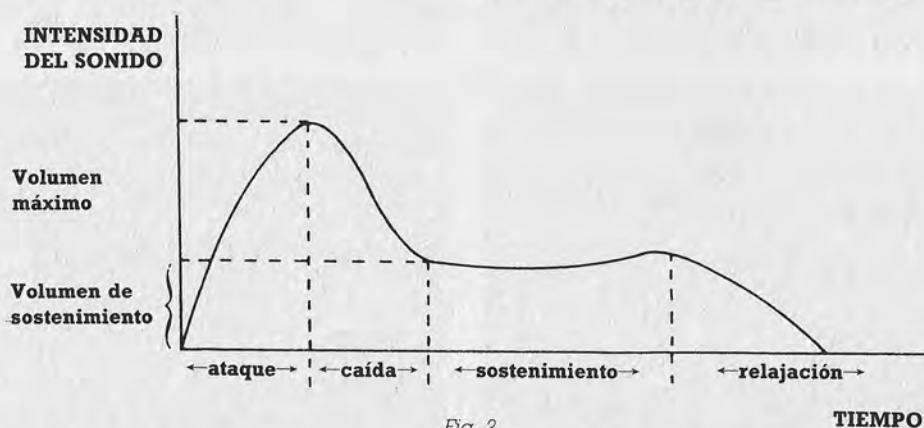


Fig. 2.

Podemos distinguir cuatro partes distintas en la curva.

1. ATAQUE = El volumen del sonido crece hasta el nivel máximo.

2. CAIDA = El volumen comienza a caer hasta un cierto punto en el cual se estabiliza.

3. SOSTENIMIENTO = Durante un cierto tiempo el volumen se sostiene y, al final, sube un poco.

4. RELAJACION = El volumen termina de caer hasta desaparecer.

Estos cuatro datos podremos introducirlos en el ordenador en una serie de registros preparados para ello. Como cada uno de estos valores puede estar comprendido entre 0 y 15, esto nos lleva a que necesitamos 4 bits para definir cada uno de estos datos. Como hay cuatro datos distintos necesitamos tener

dos registros para cada uno de los canales de sonido. Estos registros son:

Canal 1 = 5 y 6

Canal 2 = 12 y 13

Canal 3 = 19 y 20

Los valores que tenemos que introducir han de tener el siguiente formato:

POKE 54272+Registro1,A*16+C

POKE 54272+Registro2,S*16+R

donde:

A = ataque

C = caída

S = sostenimiento

R = relajación

Por ejemplo, para definir el volumen del canal de sonido 2, podríamos poner:

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

POKE 54272+12,23

POKE 54272+13,43

En la figura 3 aparecen los valores que se pueden pokear en esta dirección y el efecto que tendría en el volumen.

Valor	Tiempo de Ataque	Tiempo de Caída y Relajación
0	2 ms	6 ms
1	8 ms	24 ms
2	16 ms	48 ms
3	24 ms	72 ms
4	38 ms	114 ms
5	56 ms	168 ms
6	68 ms	204 ms
7	80 ms	240 ms
8	100 ms	300 ms
9	250 ms	750 ms
10	500 ms	1.5 s
11	800 ms	2.4 s
12	1 s	3 s
13	3 s	9 s
14	5 s	15 s
15	8 s	24 s

Fig. 2.

Una vez llegados a este punto, ya estamos en disposición de darle al ordenador las notas que queremos oír. Para ello hay que indicarle cuál es la frecuencia de la nota. Primero tenemos que encontrar lo que se llama parte alta y parte baja de la frecuencia y después pokear estas dos partes en dos registros por canal. Estos registros son:

Canal 1 = 0 Y 1

Canal 2 = 7 Y 8

Canal 3 = 14 Y 15

El COMMODORE puede abarcar hasta ocho escalas de sonido. La figura 4 nos muestra una tabla con las notas de las ocho octavas y los valores de la alta y baja frecuencia a introducir.

Nota	Octava	Alta frecuencia	Baja frecuencia
DO	0	1	18
DO sostenido	0	1	35
RE	0	1	52
RE sostenido	0	1	70
MI sostenido	0	1	90
FA sostenido	0	1	110
FA sostenido	0	1	132
SOL	0	1	155
SOL sostenido	0	1	179
LA	0	1	205
LA sostenido	0	1	133
SI	0	2	6
DO	1	2	37
DO sostenido	1	2	69
RE	1	2	104
RE sostenido	1	2	140
MI	1	2	179
FA sostenido	1	2	220
FA	1	2	8
SOL	1	3	54
SOL sostenido	1	3	103
LA	1	3	155
LA sostenido	1	2	210
SI	1	4	12
DO	2	4	73
DO sostenido	2	4	139
RE	2	4	208
RE sostenido	2	5	25

Nota	Octava	Alta frecuencia	Baja frecuencia
MI	2	5	103
FA	2	5	185
FA sostenido	2	6	16
SOL	2	6	108
SOL sostenido	2	6	206
LA	2	7	53
LA sostenido	2	7	163
SI	2	8	23
DO	3	8	147
DO sostenido	3	9	21
RE	3	9	159
RE sostenido	3	10	60
MI	3	10	205
FA	3	11	114
FA sostenido	3	12	32
SOL	3	12	216
SOL sostenido	3	13	156
LA	3	14	107
LA sostenido	3	15	70
SI	3	16	47
DO	4	17	37
DO sostenido	4	18	42
RE	4	19	63
RE sostenido	4	20	100
MI	4	21	154
FA	4	22	227
FA sostenido	4	24	63
SOL	4	25	177

Nota	Octava	Alta frecuencia	Baja frecuencia
SOL sostenido	4	27	56
LA	4	28	214
LA sostenido	4	30	141
SI	4	32	94
DO	5	34	75
DO sostenido	5	36	85
RE	5	38	126
RE sostenido	5	40	200
MI	5	43	52
FA	5	45	198
FA sostenido	5	48	127
SOL	5	51	97
SOL sostenido	5	54	111
LA	5	57	172
LA sostenido	5	61	126
SI	5	64	188
DO	6	68	149
DO sostenido	6	72	169
RE	6	76	252
RE sostenido	6	81	161

Nota	Octava	Alta frecuencia	Baja frecuencia
MI	6	86	105
FA	6	91	140
FA sostenido	6	96	254
SOL	6	102	194
SOL sostenido	6	108	223
LA	6	115	88
LA sostenido	6	122	52
SI	6	129	120
DO	7	137	43
DO sostenido	7	145	83
RE	7	153	247
RE sostenido	7	163	31
MI	7	172	210
FA	7	183	25
FA sostenido	7	193	252
SOL	7	205	133
SOL sostenido	7	217	189
LA	7	230	176
LA sostenido	7	244	103
SI	7	253	46

Fig. 4.

Finalmente, y ya para terminar, es necesario definir la forma de la onda. El COM-MODORE dispone de cuatro formas básicas. Los registros que las gobiernan según el canal de sonido son:

Canal 1 = 4
Canal 2 = 11
Canal 3 = 18

en estas posiciones pondremos un número del 1 al 4, cuyo significado es:

Valor 1 = onda triangular
Valor 2 = onda diente de sierra
Valor 3 = onda cuadrada
Valor 4 = ruido

Una vez conocido todo esto ya estamos en disposición de poder crear música con el COMMODORE. Para poder ver un ejemplo, a continuación aparecen dos programas.

```

100 REM *****
110 REM * MODULADOR DE AMPLITUD PARA COMMODORE *
120 REM *****
130 REM
140 FOR I=54272 TO 54296
150   POKE I,0
160 NEXT I
170 POKE 54296,15
180 LET A=6
190 LET D=6
200 LET R=6
210 LET S=6
220 PRINT CHR$(147)
230 PRINT "ATAQUE", "CAIDA", "SOSTEN", "RELAX", A, D, S, R
240 FOR I=1 TO 8
250   POKE 54277, A*16+D
260   POKE 54278, S*16+R
270   READ BF

```

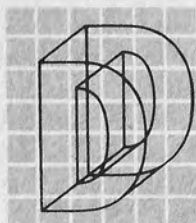
MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```
280 READ AF
290 POKE 54272 ,BF
300 POKE 54273 ,AF
310 POKE 54276 ,33
320 FOR T=1 TO 100
330 NEXT T
340 POKE 54276 ,32
350 NEXT I
360 PRINT SPC(200)
370 PRINT "PULSA UNA TECLA DE FUNCION PARA VARIAR LOS VALORES DE LA ENVOLVENTE"
380 GET A$
390 IF A$="" THEN GOTO 380
400 IF ASC(A$)<133 OR ASC(A$)>140 THEN GOTO 380
410 ON ASC(A$)-132 GOSUB 460,490,520,550,580,610,640,670
420 RESTORE
430 GOTO 220
440 DATA 75,35,126,38,52,43,198,45
450 DATA 97,51,172,57,188,64,149,68
460 LET A=A+1
470 IF A>15 THEN LET A=6
480 RETURN
490 LET D=D+1
500 IF D>15 THEN LET D=6
510 RETURN
520 LET S=S+1
530 IF S>15 THEN LET S=6
540 RETURN
550 LET R=R+1
560 IF R>15 THEN LET R=6
570 RETURN
580 LET A=A-1
590 IF A<0 THEN LET A=6
600 RETURN
610 LET D=D-1
620 IF D<0 THEN LET D=6
630 RETURN
640 LET S=S-1
650 IF S<0 THEN LET S=6
660 RETURN
670 LET R=R-1
680 IF R<0 THEN LET R=6
690 RETURN
```

```
100 REM *****
110 REM * FORMA DE ONDA EN EL COMMODORE *
120 REM *****
130 REM
140 FOR I=54272 TO 54296
150 POKE I,0
160 NEXT I
170 POKE 54296 ,15
180 POKE 54277 ,68
190 POKE 54278 ,132
200 LET FO=6
210 PRINT CHR$(147)
220 FOR I=1 TO 8
230 READ BF
```

240 READ AF
250 POKE 54272 ,BF
260 POKE 54273 ,AF
270 POKE 54276 ,FO+1
280 FOR T=1 TO 100
290 NEXT T
300 POKE 54276 ,FO
310 NEXT I
320 LET FO=FO*2
330 IF FO>128 THEN END
340 RESTORE
350 GOTO 220
360 DATA 149,68,188,64,172,57,97,51
370 DATA 198,45,52,43,126,38,75,34

TRUCOS Y RUTINAS BASICAS



ESPUES de haber visto todo lo referente a la música en el IBM, el programa que aparece a continuación nos puede servir como ejemplo de las cosas que se pueden hacer en este ordenador.

El programa desarrolla tres funciones distintas:

- Organo electrónico.
- Metrónomo.
- Compositor musical.

Estas tres funciones nos pueden servir para entender mejor cómo se puede realizar música de cierta calidad en el IBM PC y compatibles. Veamos cada una de estas partes.

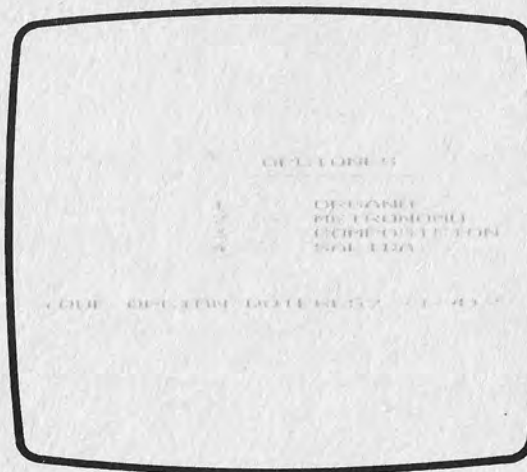


Fig. 1. Menú principal del programa.

```

1000 REM *****
1010 REM *
1020 REM *      ORGANO ELECTRONICO      *
1030 REM *      =====              *
1040 REM * VARIABLES:                  *
1050 REM *
1060 REM * NP - NUMERO DE PARTES (OPT) *
1070 REM * TP - TEMPO (ORG, MET Y INPUT) *
1080 REM * N$ - NOTA (ORG Y INPUT) *
1090 REM * LB$ - ETIQUETA DE TECLA (ORG) *
1100 REM * HZ - FRECUENCIA (MET) *
1110 REM * DR1 - DURACION DE NOTA (MET) *
1120 REM * DR2 - DURACION DE REST (MET) *
1130 REM * FL$ - FLAG DE ALTO (MET) *
1140 REM * FM$ - FLAG DE NOTA MALA (INPUT)*
1150 REM * X,Y - COORDENADAS DE CIRC.(MET)*
1160 REM * X1,2- LO MISMO *
1170 REM * NN$ - ETIQUETA DE NOTA (ORG) *
1180 REM * AN$ - ARRAY DE NOTAS (INPUT) *
1190 REM * RN - NUMERO DE FILA (INPUT) *
1200 REM * CN - NUMERAL DE COLUMN (INPUT)*
1210 REM * AL$ - FLAG DE INPUT (INPUT) *
    
```

```

1220 REM * N - NUMERO DE ELEMENT. (INP) *
1230 REM * S1$ - STRING DE MUSICA (INPUT) *
1240 REM * S2$ - LO MISMO *
1250 REM * BC$ - NOTA BUENA (INPUT) *
1260 REM * R$ - RESPUESTA (SI O NO) *
1270 REM * C$ - RESPUESTA *
1280 REM * K,L,I - INDICES *
1290 REM * *
1300 REM * LAS PALABRAS ENTRE COMILLAS SON*
1310 REM * LAS DIFERENTES PARTES DEL PRO- *
1320 REM * GRAMA. *
1330 REM *****
1340 REM
1350 KEY OFF
1360 REM
1370 REM *****
1380 REM * CABECERA *
1390 REM *****
1400 REM
1410 CLS
1420 WIDTH 40
1430 PRINT "*****"
1440 FOR I = 1 TO 19
1450 PRINT " "; TAB(40); "*"
1460 NEXT I
1470 PRINT "*****"
1480 LOCATE 8,13: PRINT "ORGANO ELECTRICO"
1490 LOCATE 9,12: PRINT "-----"
1500 LOCATE 12,7: PRINT "(c) Ed. Siglo Cultural, 1987"
1510 FOR I = 1 TO 3000
1520 NEXT I
1530 REM*****
1540 REM* ELEGIR OPCION *
1550 REM*****
1560 CLS
1570 SCREEN 0,0,0
1580 LOCATE 3,16: PRINT "OPCIONES"
1590 LOCATE 4,15: PRINT "-----"
1600 LOCATE 6,13: PRINT "1 - ORGANO"
1610 LOCATE 7,13: PRINT "2 - METRONOMO"
1620 LOCATE 8,13: PRINT "3 - COMPOSICION"
1630 LOCATE 9,13: PRINT "4 - SALIDA"
1640 LOCATE 13,3: PRINT "(QUE OPCION QUIERES? (1-4))";
1650 INPUT NP
1660 ON NP GOSUB 1700,2860,3400,1680
1670 GOTO 1550
1680 KEY ON
1690 END
1700 REM*****
1710 REM* ORGANO *
1720 REM*****
1730 CLS
1740 SCREEN 0,0,0
1750 LOCATE 4,3: PRINT "PULSA 'S' PARA TERMINAR"
1760 LOCATE 5,3: PRINT "PULSA 'N' PARA NUEVO TEMPO"
1770 PRINT:PRINT "INPUT TEMPO (32-255) - ";
1780 INPUT TP
1790 GOSUB 2160
1800 PLAY "T=" + VARPTR$(TP)
1810 PLAY "ML MB"
1820 N$ = INPUT$(1)
1830 IF N$ = "Q" THEN PLAY"O1C8":RESTORE 2650: GOSUB 2050
1840 IF N$ = "2" THEN PLAY"O1C#8":RESTORE 2660: GOSUB 2050
1850 IF N$ = "W" THEN PLAY"O1D8":RESTORE 2670: GOSUB 2050
1860 IF N$ = "3" THEN PLAY"O1D#8":RESTORE 2680: GOSUB 2050

```

TRUCOS Y RUTINAS BASICAS

```

1870 IF N$ = "E" THEN PLAY"O1E8":RESTORE 2690: GOSUB 2050
1880 IF N$ = "R" THEN PLAY"O1F8":RESTORE 2700: GOSUB 2050
1890 IF N$ = "5" THEN PLAY"O1F#8":RESTORE 2710: GOSUB 2050
1900 IF N$ = "T" THEN PLAY"O1G8":RESTORE 2720: GOSUB 2050
1910 IF N$ = "6" THEN PLAY"O1G#8":RESTORE 2730: GOSUB 2050
1920 IF N$ = "Y" THEN PLAY"O1A8":RESTORE 2740: GOSUB 2050
1930 IF N$ = "7" THEN PLAY"O1A#8":RESTORE 2750: GOSUB 2050
1940 IF N$ = "U" THEN PLAY"O1B8":RESTORE 2750: GOSUB 2050
1950 IF N$ = "I" THEN PLAY"O2C8":RESTORE 2770: GOSUB 2050
1960 IF N$ = "9" THEN PLAY"O2C#8":RESTORE 2780: GOSUB 2050
1970 IF N$ = "O" THEN PLAY"O2D8":RESTORE 2790: GOSUB 2050
1980 IF N$ = "0" THEN PLAY"O2D#8":RESTORE 2800: GOSUB 2050
1990 IF N$ = "P" THEN PLAY"O2E8":RESTORE 2810: GOSUB 2050
2000 IF N$ = "N" THEN GOTO 1730
2010 IF N$ = "S" THEN GOTO 2030
2020 GOTO 1820
2030 SCREEN 0,0,0
2040 RETURN
2050 REM
2060 REM ** BLINK KEY **
2070 REM
2080 READ X,Y,LB$
2090 LOCATE X,Y
2100 PRINT CHR$(219)
2110 FOR I = 1 TO 100
2120 NEXT I
2130 LOCATE X,Y
2140 PRINT LB$
2150 RETURN
2160 REM
2170 REM ***** DIBUJA COSAS *****
2180 REM
2190 CLS
2200 SCREEN 1,1
2210 REM
2220 REM ***** BORDE *****
2230 REM
2240 LINE (5,100) - (305,100)
2250 LINE (5,5) - (305,5)
2260 LINE (5,5) - (5,100)
2270 LINE (305,5) - (305,100)
2280 REM
2290 REM ***** TECLAS BLANCAS *****
2300 REM
2310 LET X = 35
2320 FOR I = 1 TO 9
2330 IF (I=3) OR (I=7) THEN Y=5 ELSE Y=70
2340 LINE (X,Y) - (X,100)
2350 X = X + 30
2360 NEXT I
2370 REM
2380 REM ***** TECLAS NEGRAS *****
2390 REM
2400 LET X1 = 28
2410 FOR I = 1 TO 9
2420 IF (I=3) OR (I=7) GOTO 2470
2430 LINE (X1,5) - (X1,70)
2440 X2 = X1 + 14
2450 LINE (X2,5) - (X2,70)
2460 LINE (X1,70) - (X2,70)
2470 X1 = X1 + 30
2480 NEXT I
2490 REM
2500 REM ***** ETIQUETAS DE LAS TECLAS *****
2510 REM
2520 RESTORE

```

```

2530 FOR I = 1 TO 17
2540 READ X,Y,LB$
2550 LOCATE X,Y: PRINT LB$
2560 NEXT I
2570 REM
2580 REM **** ETIQUETAS CON NOMBRE ****
2590 REM
2600 FOR I = 1 TO 10
2610 READ Y,NN$
2620 LOCATE 14,Y: PRINT NN$
2630 NEXT I
2640 RETURN
2650 DATA 11,3,"Q"
2660 DATA 8,5,"2"
2670 DATA 11,7,"W"
2680 DATA 8,9,"3"
2690 DATA 11,11,"E"
2700 DATA 11,14,"R"
2710 DATA 8,16,"5"
2720 DATA 11,18,"T"
2730 DATA 8,20,"6"
2740 DATA 11,22,"Y"
2750 DATA 8,24,"7"
2760 DATA 11,25,"U"
2770 DATA 11,29,"I"
2780 DATA 8,31,"9"
2790 DATA 11,33,"O"
2800 DATA 8,35,"0"
2810 DATA 11,37,"P"
2820 DATA 3,"C",7,"D",11,"E",14,"F",18
2830 DATA "G",22,"A",25,"B",29,"C",33
2840 DATA "D",37,"E"
2850 REM
2860 REM*****
2870 REM*          METRONOMO          *
2880 REM*****
2890 REM
2900 ON KEY(1) GOSUB 3360
2910 ON KEY(11) GOSUB 3210
2920 ON KEY(14) GOSUB 3250
2930 ON KEY(13) GOSUB 3290
2940 ON KEY(12) GOSUB 3330
2950 CLS
2960 PRINT "PULSA 'F1' PARA TERMINAR"
2970 PRINT:PRINT "DAME LA FRECUENCIA (40-8000) - ";:INPUT HZ
2980 IF (HZ > 8000) OR (HZ < 40) GOTO 2970
2990 PRINT:PRINT "DAME LA DURACION (1-5) - ";:INPUT DR1
3000 IF (DR1 < 1) OR (DR1 > 5) GOTO 2990
3010 PRINT "DAME LA PAUSA (1-25) - ";:INPUT DR2
3020 IF (DR2 < 1) OR (DR2 > 25) GOTO 3010
3030 LET N = 0
3040 SCREEN 1,1
3050 LET FL$ = "Y"
3060 KEY(1) ON
3070 KEY(11) STOP: KEY(14) STOP
3080 KEY(12) STOP: KEY(13) STOP
3090 CLS
3100 SOUND HZ, DR1
3110 X = INT(159 + 90*(COS(N)))
3120 Y = INT(99 + 90*(SIN(N)))
3130 CIRCLE ( X, Y),5,2
3140 N = N + .1
3150 KEY(11) ON: KEY(14) ON
3160 SOUND 0, DR2
3170 KEY(12) ON: KEY(13) ON
3180 IF FL$ = "Y" GOTO 3070

```

TRUCOS Y RUTINAS BASICAS

```

3190 RETURN
3200 REM ** AUMENTA LA FRECUENCIA **
3210 IF HZ > 8000 GOTO 3230
3220 HZ = HZ + 25
3230 RETURN
3240 REM ** DISMINUYE LA FRECUENCIA **
3250 IF HZ <= 65 GOTO 3270
3260 HZ = HZ - 25
3270 RETURN
3280 REM ** DISMINUYE LA DURACION DE LA PAUSA **
3290 IF DR2 <= 1 GOTO 3310
3300 DR2 = DR2 - .5
3310 RETURN
3320 REM ** AUMENTA LA DURACION DE LA PAUSA **
3330 DR2 = DR2 + .5
3340 RETURN
3350 REM
3360 LET FL$ = "N"
3370 RETURN
3380 REM
3390 REM
3400 REM*****
3410 REM*      EDITOR DE MUSICA      *
3420 REM*****
3430 REM
3440 CLS
3450 PRINT "(QUIERES INSTRUCCIONES? (S/N));"
3460 INPUT R$
3470 IF R$ = "S" GOTO 3500
3480 IF R$ = "N" GOTO 3960
3490 GOTO 3450
3500 REM ** INSTRUCCIONES **
3510 REM
3520 CLS
3530 LOCATE 3,13: PRINT "INSTRUCCIONES"
3540 LOCATE 4,12: PRINT "-----"
3550 PRINT
3560 PRINT "  PARA ESCRIBIR MUSICA NECESITAS DERLE"
3570 PRINT "LA PROGRAMMA TRES COSAS:";PRINT
3580 PRINT "      1 - QUE OCTAVA"
3590 PRINT "      2 - QUE NOTA"
3600 PRINT "      3 - QUE LONGITUD"
3610 PRINT
3620 PRINT "LAS OCTAVAS SE ESCRIBEN CON 'On'"
3630 PRINT "  - n ES LA NUMERO DE LA OCTAVA (0-6)."
3640 PRINT
3650 PRINT "LAS NOTAS SON 'ABCDEFG' CON '#' Y '-','."
3660 PRINT
3670 PRINT "LONGITUDES SON 'L' SEGUIDO DE 1,2,3,4,8 ETC."
3680 PRINT
3690 PRINT "PAUSAS SON 'P' SEGUIDO DE '1,2,4,8'"
3700 PRINT
3710 PRINT "(ENTER PARA CONTINUAR)"
3720 INPUT C$
3730 CLS
3740 PRINT
3750 PRINT "POR EXEMPLO, LA SIGUENTE LINEA TOCA UNA "
3760 PRINT "'A' EN LA OCTAVA 1, UNA 'B' EN LA OCTAVA 2,"
3770 PRINT "Y UNA 'C' EN LA OCTAVA 3. FIJATE QUE LA ULTIMA"
3780 PRINT "NOTA ES MAS LARGA."
3790 PRINT
3800 PRINT
3810 PRINT "O1A4 O2B4 O3C1"
3820 PRINT
3830 PRINT "{(RETURN ESCUCHAR)"
3840 INPUT R$

```

```

3850 PLAY "MF T100 01A4 02B4 03C1"
3860 PRINT
3870 PRINT
3880 PRINT "UNA 'L' SEGUIDA DE UN NUMERO HACE QUE TODAS"
3890 PRINT "LAS NOTAS TENGAN LA MISMA LONGITUD":PRINT
3900 PRINT "FIJATE EN EL EJEMPLO ANTERIOR"
3910 PRINT
3920 PRINT "L4 01A 02B 03C2"
3930 PRINT
3940 PRINT "(ENTER PARA CONTINUAR)"
3950 INPUT R$
3960 CLS
3970 REM
3980 REM ** ENTRADA DEL TONO **
3990 REM
4000 DIM AN$(360)
4010 FOR I = 1 TO 360
4020 LET AN$(I) = " "
4030 NEXT I
4040 CLS
4050 PRINT "INTRODUCE LAS NOTAS (X PARA FIN): "
4060 LET I = 1
4070 LET N$ = " "
4080 WHILE (N$ <> "X") AND (I <= 360)
4090 N$ = INPUT$(1)
4100 GOSUB 5110
4110 IF FL$ = "N" THEN GOTO 4080
4120 PRINT N$;
4130 AN$(I) = N$
4140 I = I + 1
4150 WEND
4160 PRINT
4170 PRINT
4180 IF I > 360 THEN LOCATE 10,5: PRINT "NUMERO MAXIMO DE NOTAS AGOTADO":PRINT "
PULSA RETURN.": INPUT R$
4190 REM
4200 REM ** EDITAR CANCION **
4210 REM
4220 CLS
4230 LOCATE 1,1: PRINT "(QUIERES CAMBIAR LA MUSICA? (S/N))";
4240 INPUT R$
4250 IF R$ = "N" GOTO 4690
4260 IF R$ <> "S" GOTO 4230
4270 CLS
4280 PRINT "USA <- Y -> PARA MOVER EL CURSOR": PRINT "USA 'F1' PARA ENTRAR UNA N
OTA NUEVA":PRINT "USA 'F1' Y 'X' A LA VEZ PARA SALIR"
4290 GOSUB 4310
4300 GOTO 4490
4310 REM
4320 REM ** IMPRESION DE LA CADENA MUSICAL **
4330 REM
4340 LET RN = 3
4350 LET CN = 1
4360 LET N = 1
4370 FOR K = 1 TO 9
4380 RN = RN + 2
4390 LOCATE RN, CN
4400 FOR L = 1 TO 40
4410 PRINT AN$(N);
4420 N = N + 1
4430 NEXT L
4440 NEXT K
4450 LET RN = 6: LET CN = 1
4460 LOCATE RN, CN: PRINT "-"
4470 AL$ = "N"
4480 RETURN

```

TRUCOS Y RUTINAS BASICAS

```
4490 REM
4500 REM ** RUTINAS DE MOVIMIENTO DEL CURSOR **
4510 REM
4520 ON KEY(13) GOSUB 5230
4530 ON KEY(1) GOSUB 5390
4540 ON KEY(12) GOSUB 5310
4550 KEY(12) ON: KEY(13) ON: KEY(1) ON
4560 IF AL$ = "N" GOTO 4520
4570 KEY(12) OFF: KEY(13) OFF: KEY(1) OFF
4580 N$ = INPUT$(1)
4590 GOSUB 5110
4600 IF N$ = "X" GOTO 4190
4610 N = (RN - 6)*20 + CN
4620 LET AN$(N) = N$
4630 LOCATE RN-1, CN: PRINT N$
4640 LOCATE RN, CN: PRINT " "
4650 CN = CN + 1
4660 LOCATE RN, CN: PRINT "- "
4670 LET AL$ = "N"
4680 GOTO 4520
4690 REM
4700 REM ** TOCAR CANCION **
4710 REM
4720 REM
4730 REM ** ENSAMBLAJE **
4740 REM
4750 S1$ = ""
4760 FOR J = 1 TO 180
4770 S1$ = S1$ + AN$ (J)
4780 NEXT J
4790 S2$ = ""
4800 FOR J = 181 TO 360
4810 S2$ = S2$ + AN$ (J)
4820 NEXT J
4830 CLS
4840 PRINT "(QUE TEMPO QUIERES? (32-255):";
4850 INPUT TP
4860 IF (TP < 32) OR (TP > 255) GOTO 4830
4870 GOSUB 4320
4880 PLAY "MF T=" + VARPTR$(TP)
4890 ON ERROR GOTO 5410
4900 PLAY "X" + VARPTR$(S1$)
4910 ON ERROR GOTO 5410
4920 PLAY "X" + VARPTR$(S2$)
4930 CLS
4940 LOCATE 7,1: PRINT "(QUIERES CAMBIAR ESTA CANCION? (S/N)";
4950 INPUT R$
4960 IF R$ = "N" GOTO 4990
4970 IF R$ <> "S" GOTO 4940
4980 GOTO 4270
4990 LOCATE 8,1: PRINT "(QUIERES TOCAR ESTA CANCION? (S/N)";
5000 INPUT R$
5010 IF R$ = "N" GOTO 5040
5020 IF R$ <> "S" GOTO 4990
5030 GOTO 4690
5040 LOCATE 9,1: PRINT "(QUIERES COMPONER OTRA CANCION: (S/N)";
5050 INPUT R$
5060 IF R$ = "N" GOTO 5090
5070 IF R$ <> "S" GOTO 5040
5080 GOTO 4010
5090 RETURN
5100 END
5110 REM
5120 REM
5130 REM
5140 RESTORE 5220
```

```

5150 LET FL$ = "N"
5160 FOR K = 1 TO 18
5170 READ BC$
5180 IF BC$ = N$ THEN FL$ = "S"
5190 NEXT K
5200 IF (FL$ = "N") AND (N$ <> "X") THEN SOUND 100, 3
5210 RETURN
5220 DATA "L", "O", "P", "A", "B", "C", "D", "E", "F", "G", "1", "2", "3", "4", "8", " ", "#", "-"
5230 REM
5240 REM ** MOVIMIENTO DEL CURSOR A LA DERECHA **
5250 REM
5260 IF (CN = 40) AND (RN = 22) GOTO 5300
5270 LOCATE RN, CN: PRINT " "
5280 IF CN < 40 THEN CN = CN + 1 ELSE LET CN = 1 : RN = RN + 2
5290 LOCATE RN, CN: PRINT "-"
5300 RETURN
5310 REM
5320 REM ** MOVIMIENTO DEL CURSOR A LA IZQUIERDA **
5330 REM
5340 LOCATE RN, CN: PRINT " "
5350 IF (CN = 1) AND (RN > 6) THEN LET CN = 40: RN = RN - 2
5360 IF CN > 1 THEN CN = CN - 1
5370 LOCATE RN, CN: PRINT "-"
5380 RETURN
5390 AL$ = "Y"
5400 RETURN
5410 REM
5420 REM ** ERROR EN EL STRING **
5430 CLS
5440 REM
5450 PRINT "HAY ERROR EN LA MUSICA."
5460 PRINT "PUEDE SER POR ":PRINT
5470 PRINT " 1 - UNA 'O' SIN NUMERO DESPUES"
5480 PRINT " 2 - UNA 'L' SIN NUMERO DESPUES"
5490 PRINT " 3 - UNA 'P' SIN NUMERO DESPUES"
5500 PRINT:PRINT "COMPRUEBA QUE TODAS LAS O's, L's Y P's"
5510 PRINT "VAN SEGUIDAS DE UN NUMERO."
5520 RESUME 4940

```

Cuando pulsamos la opción uno (1) del menú general, nos aparecerá en pantalla el teclado de un piano. En las teclas de dicho teclado podremos ver una serie de letras y números. Pulsándolas podemos obtener las notas a las que se refieren.

Si pulsamos la opción dos (2), entonces entramos en la función metrónomo. Esta puede servirnos para entender mejor el uso de la sentencia SOUND. Al entrar en esta opción el ordenador nos pedirá una serie de datos sobre la frecuencia y el tono a utilizar para después empezar a sonar el metrónomo. Los datos que le introduzcamos no son permanentes, sino que si pulsamos las teclas del cursor que se encuentran en el teclado numérico de la derecha, podremos aumentar o disminuir el tempo y la frecuencia del sonido.

Por último, al pulsar la opción tres (3) podremos componer nuestra propia música, escucharla, modificarla, etc. Con ello podremos

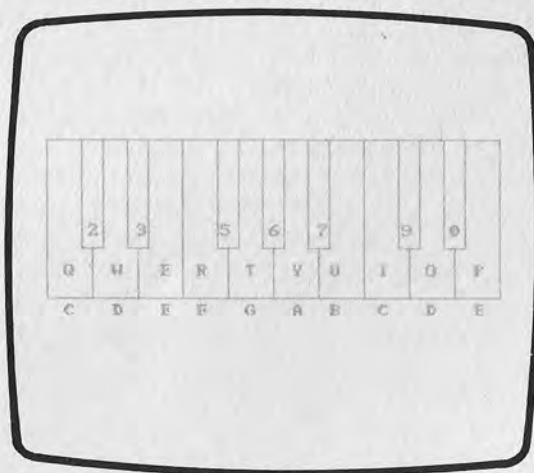


Fig. 2. El teclado de IBM como teclado de un órgano.

mos coger más práctica en el uso de la sentencia PLAY. No hace falta dar ninguna instrucción sobre el funcionamiento de esta opción, ya que aparecen todas por pantalla.

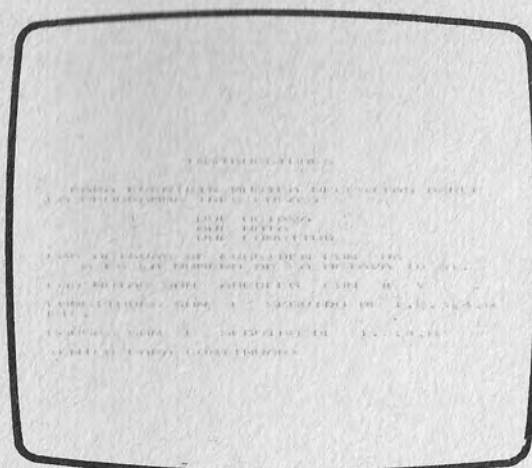


Fig. 3. Instrucciones para componer música.

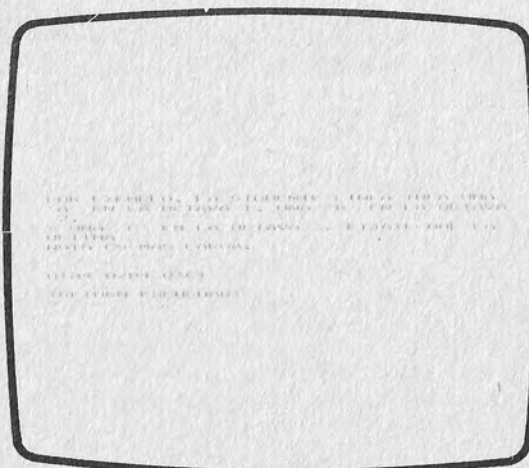
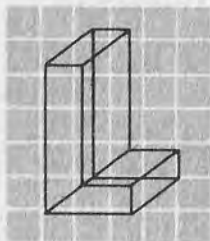


Fig. 4. Ejemplo de cómo componer música.

■ Equipo de verificación funcional de tarjetas



A fabricación de equipos electrónicos requiere en su etapa final la verificación de que cumple con las funciones para las que fue diseñado y que por tanto puede utilizarse sin problemas.

A medida que los equipos aumentan en complejidad es necesario efectuar numerosos ensayos de las posibles combinaciones que pueden darse en la operación normal por cualquiera de sus potenciales usuarios. Puede ocurrir que aunque todos los componentes utilizados se encuentren inicialmente en perfectas condiciones, el proceso de montaje haya causado el fallo de alguno de ellos o que alguna soldadura resulte defectuosa. Es también frecuente encontrar casos de componentes que presentan un comportamiento marginal y que al ser introducidos en el circuito real dejan de funcionar correctamente por interferencia con otros componentes. Para garantizar el funcionamiento correcto de los equipos, suele disponerse en el programa de control activado al arrancar una rutina de autoverificación del sistema, para ejercitar todas las funciones internas y comunicar al usuario cuanto antes que algún componente no funciona correctamente.

En la producción de equipos informáticos en gran volumen suele efectuarse una prueba de funcionamiento además de un ensayo de componentes dentro del circuito ("in circuit") para rechazar lo antes posible todos los módulos que presenten algún fallo.

La mayoría de los sistemas actuales se diseñan de forma que puedan ser adaptados a diferentes aplicaciones mediante la adición

de tarjetas en conectores de expansión colocados a tal efecto. Para que las tarjetas puedan ser intercambiables y poder montarlas en cualquier posición se definen los conectores de expansión de manera que sus conexiones se encuentren en paralelo, denominándose frecuentemente BUS al conjunto de señales que se comparten por las diferentes tarjetas del sistema. Realmente pueden existir varios BUSES en un sistema, denominándose BUS EXTERNO el dedicado a la conexión de periféricos opcionales.

Ya se presentaron en tomos anteriores las condiciones necesarias para la conexión a cada uno de los buses de los ordenadores personales de nuestro interés. En este tomo vamos a presentar el diseño de una tarjeta adaptadora que permita la verificación del funcionamiento de las tarjetas diseñadas usando el propio ordenador personal pero en condiciones más cómodas que la inserción normal en el bus, pues para ello es necesario apagar la alimentación, para evitar averías en el momento de la conexión.

En resumen, la tarjeta que se presenta permitirá el ensayo de equipos y tarjetas mediante un ordenador personal sin que el equipo utilizado en la prueba «sufra» las consecuencias de un fallo, que con seguridad se produciría si lo conectáramos en condiciones reales, además de aportar otras ventajas como, por ejemplo, la reducción del tiempo del ensayo y el registro automático de averías.

■ Necesidades a cubrir por la tarjeta

Para verificar el funcionamiento de una tarjeta que normalmente opera conectada a un bus de un ordenador personal será necesario suministrarle las señales con las que normal-

EL TALLER DEL HARDWARE

mente trabaja. El proceso de verificación consistirá en la medida del cumplimiento de las especificaciones de salida al darse las condiciones de entrada. Esta correspondencia es, en general, de tipo excitación-respuesta y con cumplimiento de una temporización muy estricta. Sin embargo, la mayoría de los fallos de los equipos y tarjetas recién montados suelen presentar una alteración del funcionamiento del tipo SI/NO en alguno de los procesos intermedios de la función. Esto quiere decir, en palabras más claras, que mediante ensayos de tipo estático o casi estático podremos eliminar aquellos equipos que resultarían defectuosos en un ensayo global, con la ventaja de que el ensayo a efectuar es mucho más simple de realizar y por tanto más económico.

Los requerimientos principales para la realización del ensayo son:

- Activación de las alimentaciones controlada por programa.

- Posibilidad de medida de los consumos de cada alimentación.

- Generación de las señales de salida del bus en todas las configuraciones posible o al menos las utilizadas en el funcionamiento real.

- Posibilidad de desactivación de las señales de salida, por paso a tercer estado o desconexión mediante relés.

- Conexión de las señales de entrada de forma que puedan ser leídas en todo momento por programa.

- Detección de cortocircuitos sin que afecten al funcionamiento del equipo que efectúa los ensayos.

- Generación de señales de temporización, controladas por programa, similares a las utilizadas en funcionamiento real.

- Generación de señales repetitivas de producción de un fenómeno, para facilitar la observación de la señal en osciloscopio.

- Posibilidad de trabajar a «reloj parado», para analizar el comportamiento de los circuitos combinatoriales conectados al bus.

- Registro de las incidencias detectadas, para posterior determinación del componente que las origina y facilitar la reparación. De fácil realización utilizando un ordenador personal.

- Señales indicativas de funcionamiento, para indicar en qué momento puede conectarse y desconectarse la tarjeta a ensayar.

Un equipo que además pretendiera co-

nocer los límites de funcionamiento de un equipo y sus prestaciones máximas debería contar con:

- Control variable sobre las tensiones y corrientes de las fuentes de alimentación.

- Capacidad de medida de frecuencias y tiempos.

- Variación del entorno de funcionamiento: temperatura, humedad, choque, ruido en las alimentaciones, etc.

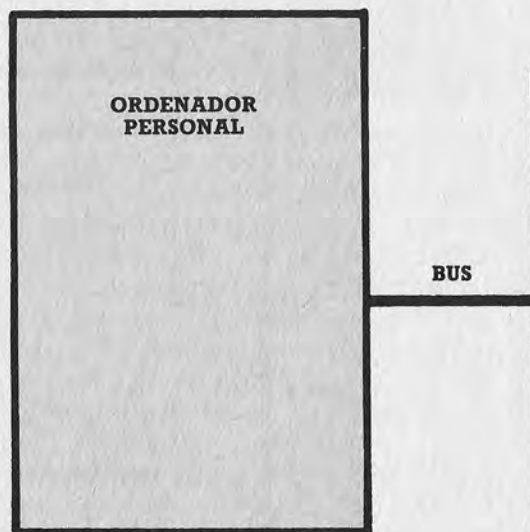


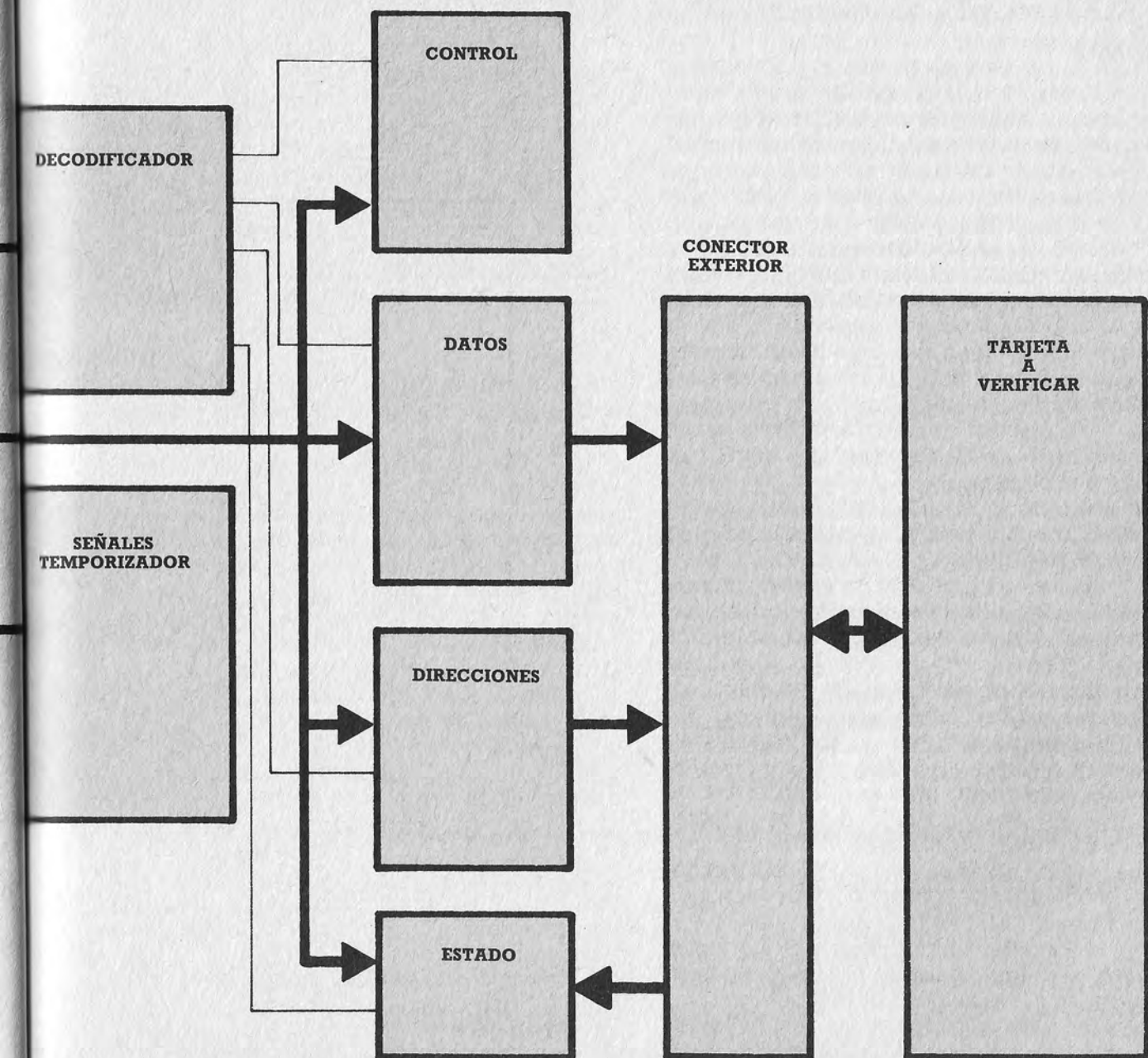
Fig. 1. Diagrama de bloques.

Generalmente este último tipo de ensayos solamente es necesario realizarlo sobre las primeras unidades o muestras de una gran producción, para garantizar que el proceso de fabricación se mantiene con la calidad adecuada. La realización de este tipo de ensayos no la contemplaremos con el montaje de nuestro diseño, que pretende solamente mostrar la potencia de cualquier ordenador personal para la verificación de sus propias tarjetas.

■ Diagrama de bloques

El sistema de ensayo de tarjetas se compondrá de:

- Bus de conexión al ordenador personal.
- Circuito de decodificación de los diferentes módulos de la tarjeta.
- Circuitos de almacenamiento de las



EL TALLER DEL HARDWARE

señales de control de los módulos externos (alimentación, indicadores, etc.).

- Circuitos de lectura de las señales indicadoras de estado.

- Circuitos de simulación de la señales de salida del bus.

- Circuitos de lectura de las señales de entrada del bus.

- Circuitos de control de las señales de temporización.

- Conector adaptado a las tarjetas a ensayar.

■ Bus de conexión al ordenador personal

El tipo de conexión necesario es el apropiado a nuestro ordenador personal que se quiera utilizar como máquina de control. Este ordenador puede ser diferente del que realmente utilicen las tarjetas a ensayar. Ni qué decir tiene que el sistema puede emplearse para el ensayo de cualquier equipo que pueda ser controlado con una configuración de señales como la indicada. Incluso podríamos verificar tarjetas de un sistema de 16 bits con un ordenador personal de 8 bits, sin pérdida de funcionalidad. Es necesario que las señales de control y de estado sean accesibles, bien directamente o a través de otros circuitos relacionados con ellas, de forma que pueda forzarse cualquier condición normal de funcionamiento y observarse el estado que alcanza el sistema. Estos requerimientos en algunos sistemas pueden condicionar el ensayo, al diseño de algún tipo de conector o pinza adicional que permita acceder a puntos del circuito no accesible directamente en el bus general.

La conexión será similar a las diseñadas para las tarjetas típicas de ampliación de puertos de entrada-salida, pues las señales de control y estado se presentarán al programa como puertos.

■ Circuito de decodificación

Se realiza con los decodificadores a partir de las direcciones generadas por el micro del ordenador. Debemos proporcionar tantas señales diferentes como puertos seleccionables existan en el sistema. Cada grupo de señales del equipo a ensayar se agruparán

en bloques de 8 bits, que se realizarán con registros de 8 bits con salida tristate. Un circuito apropiado para esta función es el 74LS374.

■ Circuitos de control

Estarán realizados por registros de salida. Para el control de las señales de alimentación o que puedan requerir grandes corrientes, se utilizará un circuito de amplificación especial para cada caso. Los circuitos apropiados son del tipo anterior o amplificadores con salida en transistor con colector abierto del tipo empleado para la activación de relés, como el 75451.

Las señales de alimentación es conveniente controlarlas a través de relés, por varias razones: la primera es porque así la caída de tensión causada por el circuito conmutador resulta despreciable y la segunda porque permite un aislamiento prácticamente total entre los circuitos de control y el que se encuentra bajo ensayo. Con el montaje indicado en la figura podría realizarse la medida de la corriente de forma automática, mediante la inserción de una pequeña resistencia en serie con el circuito alimentado, determinando por la caída de tensión el consumo real del sistema. La resistencia podría eliminarse para otros ensayos, mediante otro relé en paralelo con ella y controlada por otro bit.

Otros circuitos de control pueden realizarse simplemente con las salidas de los registros y que pueden desactivarse mediante la puesta en tercer estado. Es conveniente disponer de los circuitos más robustos posibles, para prevenir el mayor número posible de averías. La disposición de unas pequeñas resistencias en serie con cada una de las salidas garantizará un mayor grado de protección en el ensayo de circuitos defectuosos, sin afectar en la medida de comportamiento estático.

Algunas señales pueden requerir la producción de niveles no disponibles directamente con los circuitos de control; dispondremos de los circuitos de adaptación específicos para cada caso. Las señales de tipo reloj o los pulsos relacionados con la temporización del bus son de particular importancia, pues las transferencias y movimientos de datos entre los componentes se hacen con relación a transiciones de dichas señales, siendo muy importante la conservación de las secuencias relativas de los pulsos y no tanto su duración exacta. En cualquier caso es una consideración a

tener en cuenta en los ensayos con las primeras tarjetas y en la generación del programa que simule la producción de las señales.

■ Circuitos de lectura de las señales indicadoras de estado

Entenderemos por señales de estado, para cada tarjeta, todas aquellas que permitan conocer el comportamiento después de una determinada actuación. Pueden ser señales directamente accesibles en el BUS como la lectura de una memoria o la producción de una señal de petición de servicio por efecto de un funcionamiento interno. También las salidas de los decodificadores de dirección propios de la tarjeta ensayada pueden ser objeto de observación.

Para acceder a algunas de las variables internas de estado puede ser necesario el direccionamiento secuencial de varias direcciones, o la carga de un registro interno que facilite el acceso posterior. Es necesario un doble direccionamiento: para cargar los registros de control del bus con la dirección a acceder y la lectura posterior del registro accedido.

Las señales no accesibles mediante estos modos deberán ser conectadas a través de una pinza exterior a un puerto de entrada del mismo tipo que el de acceso al bus y que se leerá después de haber producido la activación de la secuencia apropiada.

■ Circuito de simulación de las señales del bus

El comportamiento de las señales del bus es, en general, de tipo síncrono produciéndose de manera determinada la secuencia de pulsos que dan origen a las diferentes operaciones. Por las características propias de los circuitos utilizados hay una limitación intrínseca de velocidad a la que puede funcionar el sistema, pero generalmente por debajo no hay limitación más que en los dispositivos de tipo dinámico, como las memorias. El propósito por tanto será la producción de la secuencia de señales simulada, a la velocidad posible mediante la escritura en puertos. Por supuesto que el funcionamiento a baja velocidad no garantiza un comportamiento correc-

to a su velocidad, pero permite detectar la mayoría de los fallos que si se producen en el ensayo con seguridad se producirán en operación real.

Una ventaja adicional es que pueden medirse los estados intermedios de un ciclo, que en funcionamiento real pueden tener una duración muy breve y, por tanto, resultar enmascarados por otras señales.

El bus de direcciones puede simularse mediante registros direccionados como puertos de salida.

El bus de datos, al presentar un funcionamiento bidireccional requiere disponer de circuitos que simulen los dos estados. Para funcionar como salida se utilizan registros con control de tercer estado y para entrada puertas de acceso a bus. Es necesario que el programa o la circuitería impidan que los dos sentidos estén activos al mismo tiempo, pues podrían producirse cortocircuitos transitorios entre puertas que simultáneamente se encuentran en estados complementarios.

Las señales de sincronización y control se simulan con registro de salida, siendo de tipo TRISTATE si es necesario operar de forma bidireccional.

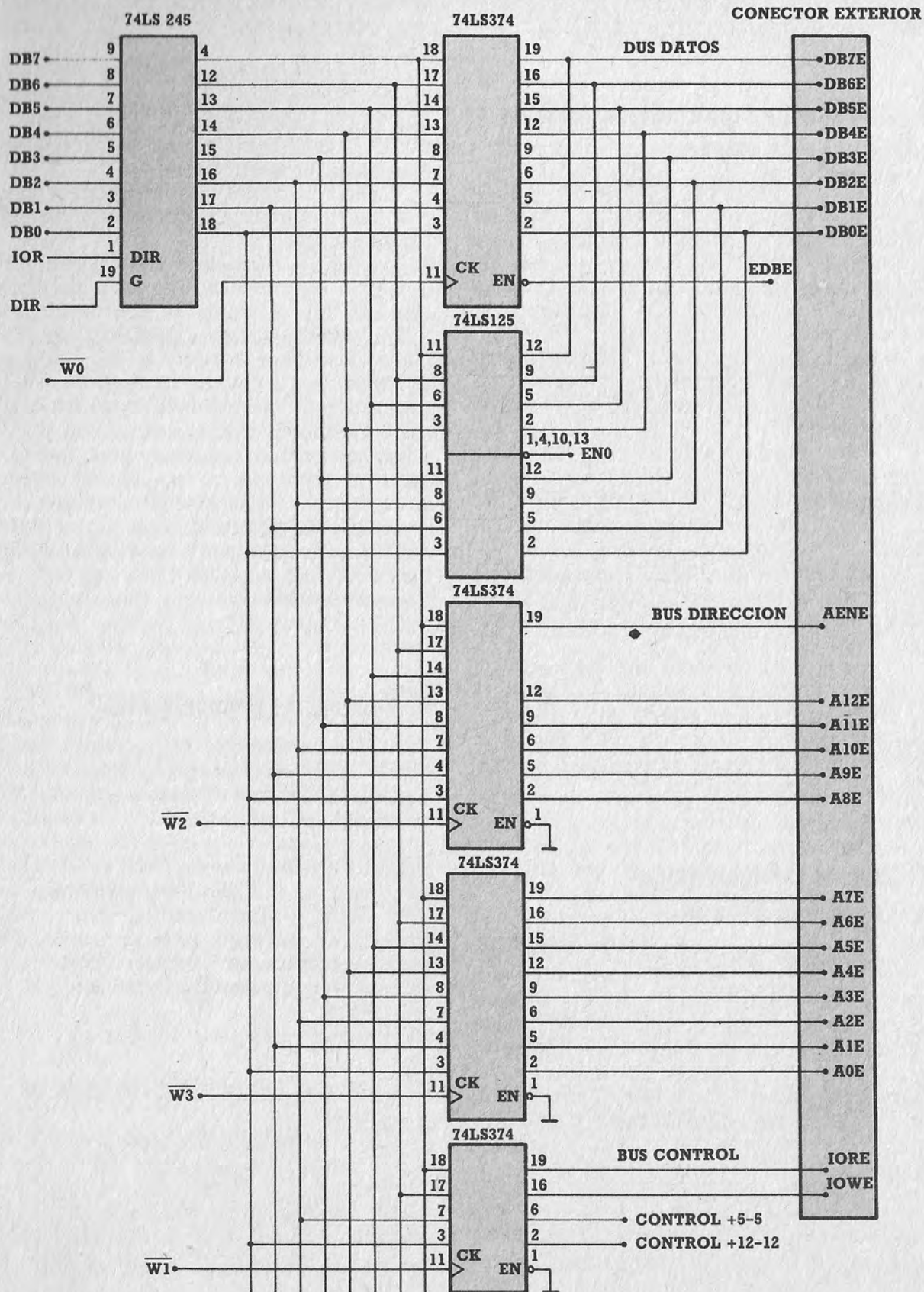
■ Señales de temporización

Los relojes que normalmente determinan el funcionamiento del sistema se simularán con registro de salida, para la producción de niveles estáticos y mediante puertas de control si es necesaria la verificación de la respuesta ante un pulso de características determinadas y que deberá generarse externamente.

Es muy importante la observación de secuencias de producción de señales que sean requeridas por la tarjeta a ensayar, para evitar el enmascaramiento de los efectos.

■ Conector adaptado a las tarjetas a ensayar

Las señales simuladas del bus se conducirán a través de un cable hasta un conector de las características propias del sistema, colocándolo en un soporte apropiado para garantizar la robustez y también la comodidad del operador.



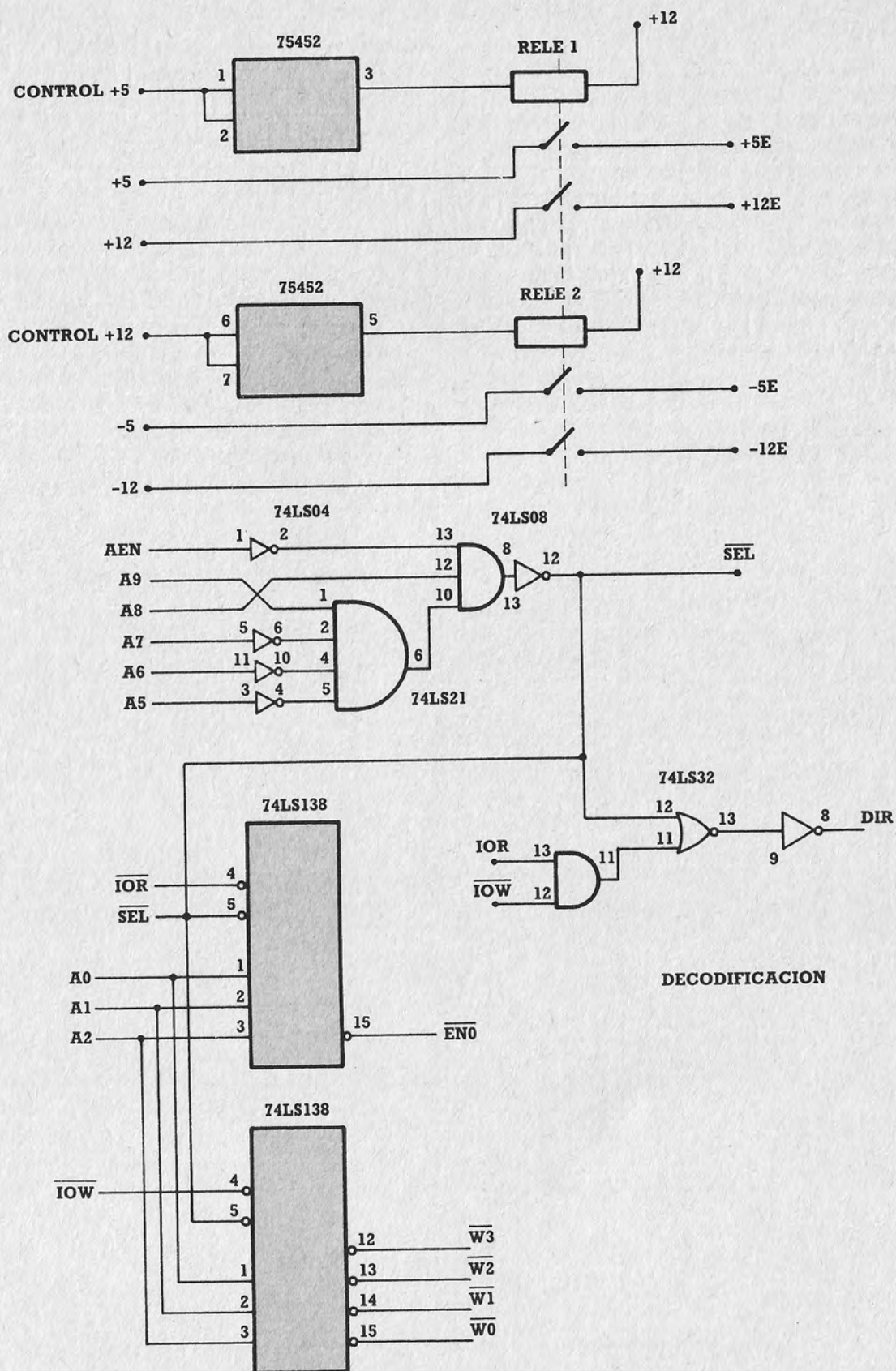


Fig. 2. Circuito de la tarjeta de verificación.

EL TALLER DEL HARDWARE

Programa de control

Para que la tarjeta realice las funciones de simulación del bus de conexión externo de la tarjeta a verificar es necesario programar el ordenador para que se produzcan las secuencias de señales apropiadas. Con el programa de ejemplo que se muestra pueden apreciarse todas las posibilidades de producción de las señales más importantes del bus, en este caso el del IBM-PC. Para adaptarlo a otras máquinas será necesario simular las señales de cada una de ellas en la misma forma que se muestra en las secuencias, por ejemplo, 6015 a 6080, donde se aprecia que con sentencias OUT y la configuración de bits indicada en la variable se generan señales equivalentes en forma a las del bus.

El programa de ejemplo muestra también cómo presentar al que realiza el ensayo, las opciones para ejercitar cada función de la tarjeta o presentar los resultados.

Conclusión

El circuito presentado ofrece la posibilidad de verificar un gran número de tarjetas y de otros equipos, que puedan activarse a través de un bus y que afortunadamente son la mayoría de los que se utilizan en los ordenadores personales. Es necesario, sin embargo, tener presente que algunas tarjetas pueden requerir un programa complejo y laborioso para poder observar todas sus funciones y, por tanto, garantizar su funcionamiento.

```
10 REM PROGRAMA DE VERIFICACION DE CIRCUITOS
20 REM UTILIZA TARJETA ADAPTADORA ESPECIAL
30 REM VERIFICA UNA TARJETA PROTOTIPO
40 REM EJEMPLO PARA EL IBM-PC
42 REM DAT% = 768 = X300 = BUS DE DATOS EXTERNO
43 REM CTRL% = 769 = X301 = -IOWE,-IORE,,, -12,+12,-5,+5
44 REM ADDH% = 770 = X302 = ...,AEN,A9,A8
45 REM ADDL% = 771 = X303 = A7,A6,A5,A4,A3,A2,A1,A0
46 ADDR0%=16:ADDR1%=17:ADDR2%=18:ADDR3%=19
57 ADDRALT%=3: CON AEN = 0
70 REM INICIALIZACION CON ALIMENTACION A 0V
80 REM Y BUS EN REPOSOS
90 V$="0123456789ABCDEF":DAT%=768:CTRL%=769:ADDH%=770:ADDL%=771
91 IORE%=128:IOWE%=64:V12%=4:V5%=1:VACTIVA%=IOWE%+IORE%+V5%+V12%
92 VDESACTIVA%=IOWE%+IORE%:REM CON DRIVER 75452
93 VACTIVAW%=IORE%+V5%+V12%:VACTIVAR%=IOWE%+V5%+V12%
94 AEN%=4:IORE%=2:ROV%=COMP%+IW%+IRGE%
96 ADMOV%=ROV%+EOV%
110 DIM AX(1000),BX(1000),CX(1000)
120 KEY OFF
130 KEY(1) ON
140 ON KEY(1) GOSUB 10000: REM DESACTIVACION DE EMERGENCIA
150 GOSUB 2000: REM TENSION INICIAL A 0
200 CLS:INPUT "IDENTIFICACION";TARJETA%
250 GOTO 9800: REM MENU PRINCIPAL
1000 REM TEST1 = ACTIVA ALIMENTACION
1001 PRINT "TEST 1, IDENTIFICACION = ";TARJETA%
1010 OUT CTRL%,VACTIVA%
1020 COMP%=56:IW%=4:IRGE%=2:GOSUB 11000: REM CONTROL
1030 DATOS2%=2:DATOS1%=255:MM%=192:GOSUB 12000: REM DATOS
1035 FOR I=1 TO 64000!
1037 X=X+1:X=0
1038 NEXT I
1040 RETURN
2000 REM DESACTIVA ALIMENTACIONES
2010 OUT CTRL%,VDESACTIVA%
2015 OUT ADDL%,255:OUT ADDH%,255
2020 RETURN
2999 REM
3000 REM TEST3 = ALIMENTACION Y FUNCION NORMAL
3001 PRINT "TEST 3, IDENTIFICACION = ";TARJETA%
3008 PRINT
3010 OUT CTRL%,VACTIVA%
3015 ERR3=0
3020 COMP%=56:IW%=4:IRGE%=2:GOSUB 11000: REM CONTROL
3022 FOR I=1 TO 255
3030 DATOS2%=1:DATOS1%=255-I:MM%=64*(I\128):GOSUB 12000: REM DATOS
```

```

3040 GOSUB 13000: REM LEE DATOS
3050 GOSUB 14000: REM MUESTRA DATOS
3060 IF A%(I)<>DATOS2% THEN 3070
3062 IF B%(I)<>DATOS1% THEN 3070
3064 IF (C%(I) AND 192)<>MM% THEN 3070
3066 GOTO 3090
3070 PRINT:ERR3=ERR3+1:PRINT "ERROR EN DATOS = ";I
3090 NEXT I
3100 OUT CTRL%,VDESACTIVAZ
3120 IF ERR3<>0 THEN PRINT "NUMERO DE ERRORES = ";ERR3:GOTO 3150
3140 PRINT:PRINT:PRINT "TODOS LOS DATOS CORRECTOS"
3150 GOSUB 20000
3500 RETURN
3999 REM
4000 REM TEST4 = ALIMENTACION Y ENVIO
4001 PRINT "TEST 4, IDENTIFICACION = ";TARJETA%
4008 PRINT
4010 OUT CTRL%,VACTIVAZ
4015 ERR3=0
4020 COMP%=56:IW%=0:IRQE%=2:GOSUB 11000: REM CONTROL
4022 FOR I=1 TO 255
4030 DATOS2%=I:DATOS1%=255-I:MM%=64*(I\128):GOSUB 12000: REM DATOS
4040 GOSUB 13000
4050 GOSUB 14000
4060 IF A%(I)<>DATOS2% THEN 4070
4062 IF B%(I)<>DATOS1% THEN 4070
4064 IF (C%(I) AND 192)<>MM% THEN 4070
4066 GOTO 4090
4070 PRINT:ERR3=ERR3+1:PRINT "ERROR ENVIANDO DATOS = ";I
4090 NEXT I
4100 OUT CTRL%,VDESACTIVAZ
4120 IF ERR3<>0 THEN PRINT "NUMERO DE ERRORES = ";ERR3:GOTO 4150
4140 PRINT:PRINT:PRINT "DATOS CORRECTOS"
4150 GOSUB 20000
4500 RETURN
4999 REM
5000 REM TEST5 = ALIMENTACION Y RECEPCION
5001 PRINT "TEST 5, TARJETA NUMERO = ";TARJETA%
5008 PRINT
5010 OUT CTRL%,VACTIVAZ
5015 ERR3=0
5020 COMP%=24:IW%=4:IRQE%=2:GOSUB 11000: REM CONTROL
5022 FOR I=2 TO 128 STEP 2
5030 ADDR%=816:DATOS2%=I:DATOS1%=128-I:MM%=0:GOSUB 12000: REM DATOS
5040 GOSUB 13000
5050 GOSUB 14000
5060 IF A%(I)<>DATOS2%\2 THEN 5070
5062 IF B%(I)<>DATOS1%\2 THEN 5070
5064 IF (C%(I) AND 192)<>MM% THEN 5070
5066 GOTO 5090
5070 PRINT:ERR3=ERR3+1:PRINT "ERROR RECIBIENDO = ";I
5090 NEXT I
5100 OUT CTRL%,VDESACTIVAZ
5120 IF ERR3<>0 THEN PRINT "NUMERO DE ERRORES = ";ERR3:GOTO 5150
5140 PRINT:PRINT:PRINT "TODO LOS DATOS CORRECTOS"
5150 GOSUB 20000
5990 RETURN
5999 REM
6000 REM TEST 6 DESACTIVA IRQ
6001 PRINT "TEST 6, IDENTIFICACION = ";TARJETA%
6002 OUT CTRL%,VACTIVAZ
6005 PRINT
6010 COMP%=56:IW%=4:IRQE%=2:GOSUB 11000: REM CONTROL
6013 OUT ADDH%,ADDRALT%
6015 FOR I=1 TO 200
6016 OUT ADDL%,ADDRO%
6017 OUT CTRL%,VACTIVAR%
6018 XXX=INP(DAT%)
6019 OUT CTRL%,VACTIVAZ
6020 OUT DAT%,ROV%
6022 OUT ADDL%,ADDR3%
6026 OUT CTRL%,VACTIVAW%
6028 OUT CTRL%,VACTIVAZ
6030 OUT DAT%,ADMOV%
6032 OUT CTRL%,VACTIVAW%
6034 OUT CTRL%,VACTIVAZ
6038 OUT ADDL%,ADDR3%
6040 OUT CTRL%,VACTIVAR%
6050 A%(I)=INP(DAT%) AND 7
6060 OUT CTRL%,VACTIVAZ

```

```

6070 GOSUB 14000
6080 NEXT I
6090 PRINT
6190 OUT CTRL%,VDESACTIVAZ%
6200 GOSUB 20000
6611 OUT ADDH%,ADDRALTAX%
6990 RETURN
6999 REM
7000 REM IMPRIME RESULTADOS
7010 FOR I=1 TO 255
7020 PRINT I;A%(I);B%(I);C%(I);"..";
7030 NEXT I
7200 GOSUB 20000
7990 RETURN
7999 REM
8000 REM TERMINACION
8010 GOSUB 2000
8080 CLS
8090 END
9000 REM AYUDA
9010 CLS
9020 PRINT "ESTE PROGRAMA PERMITE LA VERIFICACION DE UNA TARJETA DE"
9030 PRINT "PROTOTIPOS PREPARADA CON CIRCUITOS DE CONEXION A UNA RED"
9040 PRINT "EXPERIMENTAL EJERCITANDO TODAS LAS FUNCIONES DE ENVIO Y"
9050 PRINT "RECEPCION DE MENSAJES. LAS ALIMENTACIONES SE DESCONECTAN"
9060 PRINT "PARA PERMITIR LA INSERCIÓN DE UNA NUEVA TARJETA ENTRE ENSAYOS"
9070 PRINT "EL ENSAYO NO ES EN TIEMPO REAL. UNA VERSION MAS RAPIDA"
9080 PRINT "PUEDE CONSEGUIRSE UTILIZANDO EL PROGRAMA COMPILADO"
9100 PRINT
9490 INPUT "PULSE INTRO PARA CONTINUAR";A
9500 RETURN
9800 REM MENU PRINCIPAL
9810 CLS
9820 COLOR 0,7:PRINT " - PROGRAM DE VERIFICACION DE TARJETAS - ":COLOR 7,0
9830 PRINT:PRINT "F1 = DESACTIVA ALIMENTACION, IDENTIFICACION =";TARJETA%
9840 PRINT:PRINT "1.- ACTIVA ALIMENTACIONES"
9850 PRINT:PRINT "2.- DESACTIVA ALIMENTACIONES"
9860 PRINT:PRINT "3.- OPERACION NORMAL, DATOS VARIABLES"
9870 PRINT:PRINT "4.- OPERACION EXTERNA"
9880 PRINT:PRINT "5.- OPERACION INTERNA"
9890 PRINT:PRINT "6.- PRESENTA ESTADO"
9900 PRINT:PRINT "7.- PRESENTA RESULTADOS"
9910 PRINT:PRINT "8.- TERMINAR"
9920 PRINT:PRINT "9.- INSTRUCCIONES"
9930 PRINT:INPUT "PULSE OPCION ";OPCION
9940 IF (OPCION<1) OR (OPCION>9) THEN 9800
9945 CLS
9950 ON OPCION GOSUB 1000,2000,3000,4000,5000,6000,7000,8000,9000
9960 GOTO 9800
9999 REM
10000 REM DESACTIVACION DE EMERGENCIA
10010 GOSUB 2000
10020 PRINT:PRINT
10030 GOSUB 20000
10040 RETURN 9800
10999 REM
11000 REM PONE CONTROL
11020 OUT DAT%,COMP%+IW%+IRQEX%+EOV%
11030 OUT ADDL%,ADDR3%
11040 OUT ADDH%,ADDRALTAX%
11045 OUT CTRL%,VACTIVAW%
11070 OUT CTRL%,VACTIVAZ%
11500 RETURN
11999 REM
12000 ENVIA DATOS EN 816,817,818
12010 OUT DAT%,DATOS2%
12030 OUT ADDL%,ADDR0%
12035 OUT ADDH%,ADDRALTAX%
12040 OUT CTRL%,VACTIVAW%
12050 OUT CTRL%,VACTIVAZ%
12070 OUT DAT%,DATOS1%
12080 OUT ADDL%,ADDR1%
12095 OUT CTRL%,VACTIVAW%
12100 OUT CTRL%,VACTIVAZ%
12120 OUT DAT%,MM%
12130 OUT ADDL%,ADDR2%
12145 OUT CTRL%,VACTIVAW%
12170 OUT CTRL%,VACTIVAZ%
12500 RETURN
12999 REM

```

```

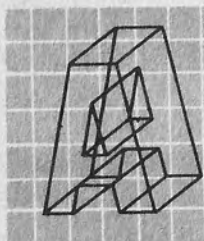
13000 REM LEE POSICION EXTERNA
13010 OUT ADDL%,ADDR0%
13030 OUT CTRL%,VACTIVAR%
13050 AX(I)=INP(DAT%)
13070 OUT CTRL%,VACTIVAR%
13110 OUT ADDL%,ADDR1%
13130 OUT CTRL%,VACTIVAR%
13150 BX(I)=INP(DAT%)
13170 OUT CTRL%,VACTIVAR%
13210 OUT ADDL%,ADDR2%
13230 OUT CTRL%,VACTIVAR%
13250 CX(I)=INP(DAT%)
13270 OUT CTRL%,VACTIVAR%
13300 RETURN
13999 REM
14000 REM PRESENTA VALOR
14010 PRINT I;:GOSUB 16000
14090 RETURN
14999 REM
15000 REM PREPARA DIRECCION
15010 ADDRALTAX=ADDR% \ 256
15020 ADDRBAJAX=ADDR% MOD 256
15030 RETURN
15999 REM
16000 REM CONVIERTE A HEX
16010 X=1+(AX(I) \ 16)
16020 Y=1+(AX(I) MOD 16)
16030 PRINT MID$(V$,X,1);MID$(V$,Y,1);", ";
16040 RETURN
16999 REM
20000 INPUT "PULSE INTRO PARA CONTINUAR";XXX
20010 RETURN

```

APRENDER CON EL ORDENADOR

NATURALEZA Y TECNOLOGIA

■ Física: cálculo de errores



Al medir una magnitud con el mismo aparato varias veces, nos damos cuenta que nos salen medidas distintas, por lo que toda medida lleva consigo un determinado error. Es necesario, por tanto, conocer la magnitud física medida, como la exactitud con la que ha sido medida, haciéndose así la física una ciencia más objetiva.

■ Fuentes y tipos de errores

De aproximación teórica

Proviene de tomar fórmulas teóricas aproximadas. Ejemplo:

$$T_1 = 2 \times \pi \times \sqrt{l/g} \rightarrow \text{fórmula aproximada}$$

$$T = 2 \times \pi \times \sqrt{l/g} \times \left(1 + \frac{1}{4} \sin^2 \theta + \dots\right) \rightarrow \text{fórmula exacta.}$$

- Error absoluto: $\Delta T = T - T_1$
- Error relativo: $\Delta T/T \times 100$
- Donde T = valor correcto
- T_1 = valor aproximado.

De aproximación numérica

Ejemplo:

$$\pi = 3.1416$$
$$\pi = 3.14159$$

Con el segundo valor se comete un error más pequeño que con el primero. Estos errores suelen ser pequeños y se pueden hacer prácticamente nulos con el ordenador.

De los propios instrumentos de medida

— Accidentales

Son debidos a pequeñas causas imposibles de controlar, que alteran los valores hallados (fluctuaciones de temperatura, presión...). Estos errores de medida se pueden aminorar hallando la media aritmética. Son errores típicamente estadísticos.

— Sistemáticos

Son debidos fundamentalmente al observador (paralelaje...) o a los instrumentos de medida (mala calidad, deformaciones con el paso del tiempo...).

Error relativo y absoluto

— Absoluto

Es la diferencia entre el valor aproximado y el valor exacto.

$$\Delta T = A - X$$

donde A es el valor aproximado y X es el valor exacto.

NOTA: Si no se conoce el valor exacto, se determina el valor medio (ver media aritmética) de la magnitud X y se toma como valor absoluto X.

Ejemplo:

$$\text{Area del círculo: } S = \pi \times r^2$$

$$S = \frac{\delta S}{\delta r} \times dr + \frac{\delta S}{\delta \pi} \times d\pi = 2 \times \pi \times r \Delta r + r^2 \pi$$

De donde se deduce que
 $\text{AREA} = S \pm \Delta S$

— Relativo

Es el cociente entre el error absoluto

(ΔX) y valor exacto (X) o el valor medio de la magnitud X.

$$Er = \Delta X / X$$

Media aritmética

Es el cociente entre la suma de las medidas efectuadas y el número de medidas realizadas.

Ejemplo:

Medidas realizadas: 2.8, 2.3, 2.85, 2.79 y 2.51

Número de medidas: 5

$$\text{media aritmética} = \frac{2.8 + 2.3 + 2.85 + 2.79 + 2.51}{5} = 2.65$$

```

10 REM*****
20 REM*      PROGRAMA      *
30 REM*      DE            *
40 REM* CALCULO DE ERRORES *
50 REM*****
60 REM* VALIDO PARA AMSTRAD *
70 REM* IBM-PC Y COMPATIBLES *
80 REM* COMMODORE, SPECTRUM *
90 REM* Y MSX              *
100 REM*****
110 CLS
120 PRINT "ESTE PROGRAMA CALCULA EL "
130 PRINT "ERROR RELATIVO Y ABSOLUTO"
140 PRINT "DE UN CONJUNTO DE MEDIDAS"
150 PRINT "A PARTIR DE LA MEDIA ARITMETICA"
160 PRINT
170 PRINT
180 LET T=0
190 DIM A(30)
200 PRINT "INTRODUCE EL NUMERO DE"
210 INPUT "MEDIDAS QUE VAS A REALIZAR:(MAXIMO 30)";N
220 IF N>20 THEN GOTO 210
230 IF N<0 THEN PRINT "ESO ES IMPOSIBLE":GOTO 210
240 FOR I=1 TO N
250 PRINT "INTRODUCE MEDIDA NUMERO ";I
260 INPUT M
270 LET A(I)=M
280 LET T=T+M
290 NEXT I
300 INPUT "PULSA ENTER PARA CONTINUAR";W$
310 CLS
320 PRINT "LA MEDIA ARITMETICA ES: ";T/N
330 LET X=T/N
340 PRINT "VA", "EA", "ER"
350 FOR J=1 TO N
360 PRINT A(J), A(J)-X, (A(J)-X)/X
370 NEXT J
380 REM *****
390 REM * MODIFICACIONES PARA COMMODORE *
400 REM *CAMBIAR LOS CLS POR :          *
410 REM * PRINT CHR$ (115)              *
420 REM *****

```

MATEMATICAS

■ Relaciones de Cardano-Vietta

Por el teorema fundamental del álgebra sabemos que un polinomio de grado -N- tiene N raíces (soluciones). Sea el polinomio genérico P(x):

$$P(x) = a_0 \times x^n + a_1 \times x^{n-1} + \dots + a_{n-1} \times x + a_n \quad (*1*)$$

Si las -N- raíces de este polinomio son: $x_1, x_2, x_3, \dots, x_n$, entonces podemos decir que:

$$P(x) = (x - x_1) \times (x - x_2) \times (x - x_3) \times \dots \times (x - x_n) \quad (*2*)$$

Si desarrollamos (*2*) e igualamos la resolución de raíces de un polinomio, llamadas las relaciones de Cardano-Vietta:

$$x_1 + x_2 + x_3 + \dots + x_n = -\frac{a_1}{a_0}$$

$$x_1 x_2 + x_1 x_3 + \dots + x_{n-1} x_n = \frac{a_2}{a_0}$$

$$x_1 x_2 x_3 + x_1 x_2 x_4 + \dots + x_{n-2} x_{n-1} x_n = -\frac{a_3}{a_0}$$

$$x_1 x_2 x_3 \dots x_n = (-1)^n \frac{a_n}{a_0}$$

Veamos la demostración con un ejemplo:

Sea el polinomio $P(x) = x^2 + 2x + 1$, tiene dos raíces x_1, x_2 .

$$\text{Entonces } P(x) = x^2 + 2x + 1 = (x - x_1)(x - x_2)$$

$$\text{si desarrollo } \Rightarrow P(x) = x^2 + 2x + 1 = x^2 - (x_1 + x_2)x + x_1 x_2$$

Si identificamos los términos de igual coeficiente $\Rightarrow$

$$1 = 1 = a_0$$

$$2 = -(x_1 + x_2) = a_1 \Rightarrow x_1 + x_2 = -a_1$$

$$1 = x_1 \times x_2 = a_2 \Rightarrow x_1 \times x_2 = a_2$$

Tenemos las relaciones de Cardano-Vietta.

Una advertencia importante es que únicamente con las relaciones de Cardano-Vietta no se pueden sacar las raíces de un polinomio, pues el sistema siempre es compatible indeterminado. Son útiles cuando conocemos al menos una relación de alguna raíz, o conocemos alguna raíz. Por ejemplo, saber que el polinomio tiene una raíz doble, una triple, que la suma de dos raíces es un determinado valor...

Ejemplo: Calcular las raíces del polinomio: $P(x) = x^3 - 3x + 2 = 0$, sabiendo que tiene una raíz doble.

RESOLUCION:

Sean las raíces $a, b, c \Rightarrow$

$$a + b + c = 0 \quad (*1*)$$

$$a \times b + a \times c + b \times c = -3 \quad (*2*)$$

$$a \times b \times c = -2 \quad (*3*)$$

Como conocemos una cuarta condición $a = b$ (raíz doble).

$$\text{de } (*1*) \Rightarrow 2 \times a + c = 0 \quad (*4*)$$

$$\text{de } (*2*) \Rightarrow a^2 + 2 \times a \times c = -3 \quad (*5*)$$

$$\text{de } (*3*) \Rightarrow a^2 \times c = -2 \quad (*6*)$$

$$\text{de } (*4*) \Rightarrow c = -2 \times a \quad (*7*)$$

de (*6*) y (*7*) sale:

$$a^2 - 4 \times a^2 = -3 \Rightarrow a = \pm 1$$

```

10 REM*****
20 REM      PROGRAMA      *
30 REM      DE            *
40 REM      RELACIONES DE *
50 REM      CARDANO-VIETTA *
60 REM*****
70 REM      VALIDO PARA AMSTRAD *
80 REM      IBM-PC Y COMPATIBLES *
90 REM      COMMODORE, SPECTRUM *
100 REM      Y MSX          *
110 REM*****
120 CLS
130 PRINT "ESTE PROGRAMA CALCULA LOS COEFICIENTES"
140 PRINT "DE LAS RELACIONES DE CARDANO-VIETTA"
    
```

```

150 PRINT "PARA ELLO DEBES INTRODUCIR EL ORDEN"
160 PRINT "DEL POLINOMIO, SIEMPRE QUE SEA MENOR"
170 PRINT "DE 20: SEGUIDAMENTE EL ORDENADOR PEDIRA"
180 PRINT "LOS COEFICIENTES UNO A UNO"
190 PRINT :PRINT
200 INPUT "PULSA <ENTER> PARA CONTINUAR";W$
210 CLS
220 DIM A(21)
230 INPUT "INTRODUCE ORDEN DEL POLINOMIO";N
240 IF N>20 THEN GOTO 230
250 FOR J=0 TO N
260 PRINT "COEFICIENTE A";J;" = ";
270 INPUT A
280 LET A(J+1)=A
290 NEXT J
300 PRINT "P(X)=";
310 FOR H=1 TO N
320 PRINT A(H);"X{";N-H+1;"} + ";
330 NEXT H
340 PRINT A(N+1);
350 PRINT :PRINT
360 INPUT "PULSA <ENTER> PARA CONTINUAR";W$
370 CLS
380 FOR J=1 TO N
390 PRINT "RELACION DE CARDANO NUMER.";J;"=";{"(-1)^J;"{";A(J+1);"/";A(1);"}=";A(J
+1)/A(1)
400 NEXT J
410 REM*****
420 REM* MODIFICACIONES PARA COMMODORE*
430 REM* CAMBIAR DONDE PONE CLS *
440 REM* PRINT CHR$(115) *
450 REM*****

```

SOCIEDAD

■ El adjetivo

El adjetivo es una parte variable de la oración que puede ir acompañando al nombre para constituir el grupo o sintagma nominal: *La excursión agradable, el viento suave, el traje azul, la habitación limpia...*

El adjetivo significa cualidades de los objetos; y esa significación la podemos graduar:

- Una excursión muy agradable.
- Una excursión poco agradable.
- Una excursión nada agradable.
- Una persona nada simpática.
- Una persona poco simpática.

Para graduar la significación del adjetivo, utilizamos palabras como los adverbios modificadores de grado (*nada, poco, muy...*), y locuciones adverbiales de grado (*en extremo, por demás, etc.*). Por tanto, la gradación del adjetivo se basa en modificaciones de la significación del mismo.

Existen solamente tres casos posibles de graduar la significación del adjetivo:

— Positivo: El adjetivo se presenta sin modificaciones de significado, su cualidad se presenta pura, ni aumentada ni disminuida, ni comparada (*La habitación limpia*).

— Comparativo: Cuando la cualidad se compara (*La habitación está más limpia que el salón*).

— Superlativo: Cuando la cualidad se presenta en su grado más alto (*La habitación está muy limpia*).

El grado comparativo a su vez puede ser de tres clases:

— Comparativo de igualdad: *Es tan pesado como el plomo.*

— Comparativo de superioridad: *Tiene la cabeza más dura que el hierro.*

— Comparativo de inferioridad: *Manuel es menos inteligente que su hermano.*

El grado superlativo cuantifica el significado del adjetivo con la máxima intensidad. Tiene dos maneras de expresarse:

— Con el sufijo -ísimo: *azulísimo, cortísimo, feísimo.*

— Con los adverbios muy, harto, bien: *muy listo, bien guapo, harto lento.*

También producen gradación superlativa otros adverbios: *excepcionalmente guapo, terriblemente precoz, extremadamente listo, especialmente cariñoso...*

APRENDER CON EL ORDENADOR

Nuestra lengua no cesa de crear procedimientos para intensificar superlativamente al adjetivo: *pedazo de burro, más que tonto*, etc.

Muchos adjetivos cuando reciben el sufijo -ísimo, recuperan su forma latina distinta de la forma simple castellana:

- de amigo, amicísimo (y amiguísimo).
- de bueno, bonísimo (y buenísimo).
- de fuerte, fortísimo o (fuertísimo).

A los ejemplos anteriores se les denomina superlativos cultos.

■ Superlativo absoluto y superlativo relativo

Con el superlativo absoluto se exalta la cualidad del objeto sin tener en cuenta ningún otro objeto:

- *Ella es listísima.*
- *El pantalón es cortísimo.*
- *Fue atrozmente cruel.*
- *Juan es extremadamente cruel.*
- *Mi perra está muy cariñosa.*
- *Manolo es listísimo.*

Con el superlativo relativo se exalta la cualidad del objeto persona, teniendo en cuenta otros objetos y personas.

■ Adjetivos especificativos y explicativos o epítetos

El adjetivo especificativo suele ir normalmente detrás del nombre y selecciona a éste dentro del grupo al que pertenece:

- *Comió con su amigo mejor.*
- *Atacaron los soldados valientes.*
- *Fue al cine con su hermano pequeño.*
- *El se comió el tomate grande.*

Los adjetivos mejor, valiente, pequeño y grande, nos precisan de qué amigo, de qué soldados, de qué hermano y de qué tomate se trata, diferenciándolo así del resto de los amigos, soldados, hermanos y tomates. El suprimir un adjetivo especificativo de una oración cambiaría el significado de esa oración.

El adjetivo explicativo o epíteto va delante del nombre y solamente nos dice una cualidad del nombre, sin diferenciarlo. Es un adjetivo que no resulta imprescindible para la exactitud significativa de la oración, si le añade viveza y color:

- *Blanca nieve.*
- *Atacaron los valientes soldados.*
- *Cogió mi húmeda toalla.*
- *Es una gran mentira.*
- *Era el manso cordero.*
- *Azul cielo.*
- *Fui por el largo camino.*

```
10 REM*****
20 REM*      PROGRAMA      *
30 REM*      DE            *
40 REM*      ADJETIVOS     *
50 REM*****
60 REM*      VALIDO PARA AMSTRAD *
70 REM*      IBM-PC Y COMPATIBLES *
80 REM*      COMMODORE, SPECTRUM *
90 REM*      Y MSX         *
100 REM*****
110 CLS
120 PRINT "ESTE PROGRAMA ANALIZA TUS CONOCIMIENTOS"
130 PRINT "DE ADJETIVOS. EL ORDENADOR TE IRA"
140 PRINT "PREGUNTANDO LOS TIPOS DE LOS ADJETIVOS"
150 PRINT "DE UNAS FRASES Y LES TENDRAS QUE RESPONDER"
160 PRINT "DE ACUERDO CON LA TEORIA"
170 PRINT
180 PRINT "POR EJEMPLO: "
190 PRINT
200 PRINT "PREGUNTA: FUE UNA EXCURSION AGRADABLE"
210 PRINT "LA RESPUESTA TENDRA QUE SER:"
220 PRINT "POSITIVO ESPECIFICATIVO"
230 PRINT
240 PRINT "PREGUNTA: FUE UNA AGRADABLE EXCURSION"
```

```

250 PRINT "LA RESPUESTE TENDRA QUE SER: "
260 PRINT "POSITIVO EPITETO"
270 PRINT :PRINT
280 INPUT "PULSA ENTER PARA CONTINUAR";W$
290 CLS
295 LET N=0
300 FOR J=1 TO 22
310 READ A$
320 PRINT "* PREGUNTA: ";A$
330 LINE INPUT "** LA RESPUESTA ES: ";R$
340 READ A$
350 IF A$=R$ THEN PRINT "CORRECTO":LET N=N+1
360 IF A$<>R$ THEN PRINT "INCORRECTO, LA RESPUESTA ERA: ";A$
370 NEXT J
380 PRINT :PRINT :PRINT
390 PRINT "LA PUNTUACION OBTENIDA ES: ";N
1000 DATA "ES UN GRAN AMIGO","POSITIVO EPITETO","ES UN AMIGO GRANDE"
1010 DATA "POSITIVO ESPECIFICATIVO","MANUEL ES MAS BAJO QUE SU HERMANO"
1020 DATA "COMPARATIVO DE INFERIORIDAD","ES UN PERRO FORTISIMO"
1030 DATA "SUPERLATIVO CULTO ESPECIFICATIVO","ES BUEN HOMBRE","POSITIVO EPITETO"
1040 DATA "ELLA ES LISTISIMA","SUPELATIVO ABSOLUTO","ESTOY HARTO CANSADO"
1050 DATA "SUPERLATIVO ABSOLUTO","ANDAMOS SOBRE LA BLANDISIMA ARENA"
1060 DATA "SUPELATIVO EPITETO","SE PEGA LA GRAN VIDA","POSITIVO EPITETO"
1070 DATA "ES MAS LISTO QUE EL HAMBRE","COMPARATIVO DE SUPERIORIDAD"
1080 DATA "LA ARENA BLANDA QUEMA","POSITIVO ESPECIFICATIVO"
1090 DATA "ERA MAS TONTO QUE ABUNDIO","COMPARATIVO SUPERIORIDAD"
1100 DATA "EL ES UN POBRE INGENUO","POSITIVO EPITETO","EL ES UN INGENUO POBRE"
1110 DATA "POSITIVO ESPECIFICATIVO","ES TAN BLANCO COMO LA NIEVE"
1120 DATA "COMPARATIVO IGUALDAD","EN BUENA HORA HAS VENIDO"
1130 DATA "POSITIVO EPITETO","ES EL PICO MAS ALTO","SUPERLATIVO ABSOLUTO"
1140 DATA "ESTE CAMINO ES MENOS LARGO QUE AQUEL","COMPARATIVO INFERIORIDAD"
1150 DATA "ES TAN BUENO COMO EL PAN","COMPARATIVO IGUALDAD"
1160 DATA "ERES UN HOMBRE INTEGRO","POSITIVO ESPECIFICATIVO"
1170 DATA "PERDIDOS EN LA FRIA NOCHE","POSITIVO EPITETO","TENEMOS UN PERRAZO"
1180 DATA "SUPERLATIVO ABSOLUTO"
2000 REM*****
2010 REM* MODIFICACIONES PARA COMMODORE *
2020 REM* CAMBIAR DONDE PONGA CLS POR: *
2030 REM* PRINT CHR$(115) *
2040 REM*****

```

PARA LOS MAS JOVENES

■ Arquitectura básica de un ordenador

Actualmente todo el mundo habla de los ordenadores, la tecnología, la informática, pero realmente utilizamos conceptos que no sabemos generalmente lo que significan. Nos resultaría muy difícil saber qué es realmente, cómo está hecho, nuestro Spectrum o nuestro Amstrad. En este capítulo vamos a introducirnos en el interesante tema de la estructura de

un ordenador, cómo está hecho, qué es lo que se llama arquitectura de un ordenador.

Por estar dedicado este tema a los más jóvenes, suponemos que no tienen conocimientos en electrónica, por lo que vamos a hacer un estudio que se denomina «de bloques», sin profundizar de momento en el apasionante mundo de la electrónica de los ordenadores.

Este programa te presentará una serie de pantallas donde te indicará qué es cada bloque y para qué sirve. Pretende ser simplemente tu primer contacto con el tema de la arquitectura interna de los ordenadores, por lo que si deseas ampliar conocimientos, no tienes más que acudir a cualquier libro básico de introducción al mundo de los ordenadores.

```

10 REM*****
20 REM*      PROGRAMA      *
30 REM*      DE            *
40 REM*  ARQUITECTURA BASICA *
50 REM*    DE UN ORDENADOR  *
60 REM*****
70 REM*  VALIDO PARA AMSTRAD *
80 REM*  IBM-PC Y COMPATIBLES *

```

APRENDER CON EL ORDENADOR

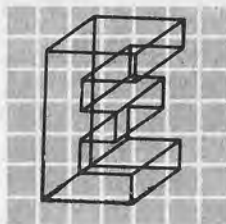
```

90 REM*   COMMODORE, SPECTRUM *
100 REM*   Y MSX *
110 REM*****
120 CLS
130 PRINT "ESTE PROGRAMA TE MUESTRA"
140 PRINT "LA ARQUITECTURA BASICA DE"
150 PRINT "UN ORDENADOR CUALQUIERA."
160 PRINT
170 INPUT "PULSA ENTER PARA CONTINUAR";A$
180 GOSUB 260
190 PRINT "1.- ORDENADOR"
200 PRINT "2.- PERIFERICOS"
210 PRINT "3.- FINAL"
220 INPUT "ELIGE OPCION NUMERO ";H
230 ON H GOSUB 380,570,1430
240 GOTO 180
250 END
260 REM *****MENU PRIMERO*****
270 CLS
280 PRINT :PRINT :PRINT
290 PRINT "*****"
300 PRINT " * ORDENADOR *"
310 PRINT "*****"
320 PRINT :PRINT
330 PRINT "*****"
340 PRINT " * PERIFERICOS *"
350 PRINT "*****"
360 PRINT :PRINT
370 RETURN
380 REM***SUB ORDENADOR****
390 GOSUB 260
400 PRINT "ORDENADOR:"
410 PRINT "ENTENDEMOS POR ORDENADOR UNA MAQUINA POTENTE"
420 PRINT "Y FLEXIBLE, QUE MEDIANTE CIRCUITOS ELECTRONICOS"
430 PRINT "LOGICOS ES CAPAZ DE EJECUTAR UNA SECUENCIA DE"
440 PRINT "OPERACIONES LLAMADAS -PROGRAMA-"
450 PRINT
460 PRINT "HAY QUE MATIZAR QUE EL ORDENADOR PROPIAMENTE DICHO"
470 PRINT "SOLO LO COMPONEN LOS CIRCUITOS ELECTRONICOS INTERNOS"
480 PRINT "Y QUE APARATOS COMO EL MONITOR Y EL TECLADO SON "
490 PRINT "EXTERIORES A ESTE DISPOSITIVO"
500 PRINT
510 INPUT "PULSA ENTER PARA CONTINUAR";W$
520 GOSUB 260
530 PRINT :PRINT
540 INPUT "DESEAS MAS INFORMACION SOBRE EL ORDENADOR (S/N) ";C$
550 IF C$="S" THEN GOSUB 800
560 RETURN
570 REM***PERIFERICOS****
580 CLS
590 PRINT "*****"
600 PRINT " * MONITOR * * IMPRESORA *"
610 PRINT "*****"
620 PRINT :PRINT
630 PRINT "*****"
640 PRINT " * TECLADO *"
650 PRINT "*****"
660 PRINT :PRINT
670 PRINT "MONITOR: EL MONITOR ES EL PERIFERICO A TRAVES"
680 PRINT "DEL CUAL EL ORDENADOR SE COMUNICA CON NOSOTROS"
690 PRINT "TECLADO: EL TECLADO SIRVE PARA INTRODUCIR EN EL"
700 PRINT "ORDENADOR LOS DISTINTOS DATOS Y PROGRAMAS"
710 PRINT "IMPRESORA: LA IMPRESORA SIRVE PARA FACILITARLOS"
720 PRINT "INFORMACION ESCRITA QUE NO DESAPAREZCA CUANDO "
730 PRINT "APAGEMOS EL ORDENADOR. "
740 PRINT "OTROS: EXISTEN MUCHOS MAS PERIFERICOS, POR EJEMPLO:"
750 PRINT "JOYSTICKS, PARA JUGAR, MODEMS PARA COMUNICARNOS POR"
760 PRINT "TELEFONO, CASSETTES Y DISCOS, PARA ALMACENAR "
770 PRINT "INFORMACION RECUPERABLE..."
780 INPUT "PULSA ENTER PARA CONTINUAR";;W$
790 RETURN
800 REM***EL ORDENADOR POR DENTRO****
810 CLS
820 GOSUB 890
830 PRINT "1.- U.L.A: UNIDAD ARITMETICO LOGICA"
840 PRINT "2.- C.P.U: UNIDAD CENTRAL DE PROCESOS."
850 PRINT "3.- M.P : MEMORIA PRINCIPAL."
860 PRINT "4.- VOLVER AL MENU PRINCIPAL"
870 INPUT "PULSA EL NUMERO QUE DESEES";A
880 ON A GOTO 1030,1150,1300,560

```


PEQUEÑA HISTORIA DE LA INFORMATICA

■ Primeros pasos hacia el multiproceso



N los ordenadores que debían procesar grandes volúmenes de datos, y en particular aquellos especializados en tareas de tipo científico, ha sido una obsesión constante obtener mejores tiempos de proceso. Desde

el año 1954 se ha seguido una carrera loca para alcanzar cotas de tiempo aceptables, para que dichas tareas tuvieran algún sentido. Pensemos de nuevo en un ordenador de un instituto meteorológico que realizara predicciones del tiempo muy fiables, teniendo en cuenta innumerables factores, pero que diera los resultados después que se produjeran los acontecimientos. Estos ordenadores sólo servirían para realizar comprobaciones de la fiabilidad de los procesos de predicción. Y, naturalmente, no tendrían sentido.

Los ordenadores que trabajan en tiempo real necesitan resolver este problema. Actualmente, para tareas que necesiten procesar multitud de datos en tiempo real, o en tiempos muy cortos por alguna otra razón, se utilizan ordenadores paralelos, que disponen de una arquitectura diferente de la de von Neumann (aunque todos los ordenadores de este tipo disponen también de un procesador con arquitectura tradicional, para tratar otras tareas que siempre suelen darse conjuntamente con las típicas propias de ordenadores paralelos).

En 1954 se dispuso por primera vez de ordenadores en multiproceso. El multiproceso consiste esencialmente en que varias unidades, o máquinas, ejecuten simultáneamente uno o varios programas, disponiendo todas

ellas de acceso a ciertos recursos comunes (como puede ser, por ejemplo, una memoria de masa enorme). En el National Bureau of Standards de los Estados Unidos se unieron por esas fechas dos ordenadores. La idea era que uno de ellos pudiera actuar de entrada/salida hacia el otro. ¡Y había nacido el multiproceso!

La idea fue todo un éxito, y dos años después la IBM fabricó una máquina con posibilidad de ser utilizada en multiproceso. Efectivamente, el modelo 305 se había diseñado para que pudiera conectarse con otro 305, disponiendo ambos de una memoria de masa común, en concreto una RAMAC. Cada ordenador disponía de dispositivos de lectura/escritura propios. Naturalmente, estaba previsto que no pudiera accederse al mismo tiempo a una posición de memoria por los dos dispositivos de lectura/escritura.

Sin embargo, no fueron los ordenadores de IBM los más conocidos por sus cualidades de multiproceso: tanto el Gamma 60, de Bull, francés, como el LARC, de Univac, americano, disponían de varios procesadores que compartían recursos comunes. Generalmente, el recurso más corriente a compartir solía ser la memoria.

El ordenador LARC había sido diseñado pensando en el multiproceso. No se trataba simplemente de que se pudiera unir a otro ordenador para poder realizar algunas tareas en multiproceso. Además, el multiproceso se ejecutaba en dos niveles. En primer lugar, el LARC disponía de una unidad de entrada/salida que llevaba a cabo estas tareas, mientras el procesador central, a su vez, procesaba otros datos, o realiza otros trabajos. El segundo nivel de multiproceso tenía lugar en el pro-

PEQUEÑA HISTORIA DE LA INFORMATICA

pio procesador central, que estaba realmente formado por dos procesadores que funcionaban independientemente, repartiéndose entre ellos las tareas. Naturalmente, todo este proceso de "reparto de tareas" era absolutamente automático, y el programador no necesitaba intervenir para nada. Con esta disposición es claro que la velocidad se había incrementado enormemente.

Univac sólo consiguió vender dos modelos de LARC a la Marina y al Ejército de los Estados Unidos. El problema con el que siempre chocaba es que era al ser máquina demasiado innovadora, no estaba muy probada, y los distintos organismos y empresas no confiaban demasiado en ordenadores que no tuvieran al menos una "pequeña historia". Sin embargo, el LARC era un magnífico ordenador, y junto con el Gamma 60, de Bull, fueron la punta de lanza de la tecnología informática de los primeros años sesenta.

Efectivamente, la verdadera competencia de la compañía Univac en ordenadores multiproceso de la segunda generación fue la Bull. El ordenador Gamma 60 era una máquina multiproceso muy conseguida, y proporcionó contratos y beneficios excelentes, especialmente con el Departamento de Defensa de los Estados Unidos. Esta máquina era capaz de procesar varios programas independientemente, todo un avance en el multiproceso. Veamos sucintamente la estructura de la máquina:

Disponía de un sistema de multiproceso mediante unidades funcionales independientes. Entre ellas, las unidades de entrada/salida, que controlaban estas tareas (preguntas directas por consola, control de lectoras de tarjetas y de cinta, perforadoras de papel y de cinta, etc.). A continuación, las unidades de proceso, una lógica y otra aritmética. La unidad lógica se ocupaba de procesar los códigos de instrumentos e instrucciones lógicas. La unidad aritmética realizaba los cálculos aritméticos. Como el ordenador utilizaba un sistema en base 10, la unidad disponía de un sistema muy original de realizar las comprobaciones, calculando los restos módulo 7. Es importante recordar que todas estas unidades que hemos mencionado eran absolutamente autónomas, es decir, podían funcionar independientemente, suponiendo un ahorro de tiempo considerable. Veamos ahora su memoria.

En primer lugar, cada unidad disponía de su propia memoria, ya que de otro modo,

no sería totalmente autónoma. Sin embargo, esta memoria podía intercambiar información con otra memoria principal, mucho más grande, para proseguir con su proceso. La memoria principal, a su vez, estaba formada por ocho bloques cada uno de ellos de cuatro mil palabras. Estaba organizada de forma muy semejante a las memorias clasificadas en bytes, pero en este caso, las palabras eran de 27 bits, de los cuales se utilizaban tres para tareas de control. La organización se basaba en unas estructuras de 24 bits (en el lugar de los bytes de ocho bits). Estas estructuras se denominaban "cadenas", y podían representar un número binario de 24 bits, o seis dígitos decimales, o caracteres de 4 a 6 bits, de un alfabeto no muy completo de 64 caracteres.

Esta memoria era muy especial, ya que era capaz de almacenar la información general que contenía, más la que le iban introduciendo las distintas unidades multiproceso durante una sesión cualquiera de trabajo.

El Gamma 60, de Bull, disponía también de un sistema traductor, cuya misión consistía en realizar las traducciones de los códigos, además de ocuparse de organizar la información y de la presentación (organización de las páginas, etc.).

Naturalmente, un equipo tan complejo necesitaba un dispositivo que coordinara la actividad de los distintos procesadores, memoria, etc. Para ello disponía de dos líneas de bus llamadas "canales" por los diseñadores del equipo. Una de las líneas tomaba la información de las unidades funcionales y las enviaba a las unidades centrales de la máquina, es decir, a los circuitos de coordinación o a la memoria central, y la segunda línea, el camino inverso, de las unidades centrales a las unidades funcionales. El esquema podría ser así: las unidades funcionales piden acceso a la memoria principal, y su petición es atendida o pospuesta, según unos mecanismos especiales de selección. Estos, desde el punto de vista del hardware de la máquina, dan prioridad a las unidades más rápidas, como norma general, dejando en espera siempre a las más lentas (que suelen ser unidades de entrada o salida). Desde el punto de vista del software, el programador indicará en los distintos programas las prioridades.

Veamos algunos inconvenientes del ordenador Gamma 60, de Bull.

Como hemos visto en anteriores líneas, el Gamma 60 era un ordenador revolucionario en muchos aspectos. Su puesta a punto, por

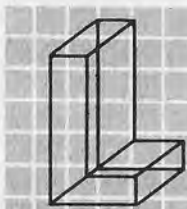
tanto, requirió más tiempo que la de otros ordenadores, algunas de cuyas unidades eran prácticamente copias o al menos muy parecidas a otras de ordenadores muy probados. La compañía Bull había previsto lanzar la máquina al mercado en el año 1959, pero dificultades de todo orden le impidieron hacerlo, saliendo el ordenador a la venta en el año 60. Este dato es importante, ya que no se trata sólo de recuperar la inversión mucho más tarde, sino que en ese tiempo las otras empresas fabricantes han avanzado en la creación de otros productos. No hay que olvidar que en campos que cambian con rapidez, el éxito de un determinado producto depende muchísimo del momento de su aparición en el mercado.

El estado francés podía haber hecho mucho por la compañía Bull, que era una de las pocas compañías europeas fabricantes de ordenadores, y, desde luego la única francesa de un cierto prestigio a nivel internacional. El desarrollo del Gamma 60 produjo muchos quebraderos de cabeza al equipo de administración de la Bull, que no disponía en esa época de financiación suficiente para abordar (como debía haber hecho) el desarrollo del ordenador, un buen sistema operativo y un lenguaje adecuado. Es probable que si el gobierno francés hubiera ayudado a la Bull, la primacía norteamericana en materia informática no hubiera sido tan evidente en los años que siguieron. Sin embargo, es muy fácil ver a posteriori las soluciones a los problemas.

El lenguaje que utilizaba el Gamma 60 era prácticamente un ensamblador. Sin embargo, a principios de los años sesenta ya se habían creado lenguajes de un nivel mucho más alto, y la mayoría de casas fabricantes vencían sus máquinas con posibilidades para programar en estos lenguajes. El Gamma 60 no disponía de compiladores de ALGOL o COBOL. Y este era un problema importante, más aún, dada la complejidad de la máquina, que además requería controlar muy bien con anterioridad los procesos que se realizarían en multiproceso y aquellos cuya ejecución sería más sencilla por el sistema tradicional. Este estudio previo de los cálculos era un engorro muy molesto, complicado además por una programación tan próxima al lenguaje máquina. La compañía Bull se asoció con A. Perlis para crear los compiladores de ALGOL y COBOL, que aparecieron dos años más tarde (1962) en el mercado.

Otro de los inconvenientes del ordenador Gamma 60 era el enorme consumo de potencia que necesitaba. Ya sabemos, de los ordenadores de la primera generación, que los grandes consumos de potencia solían estar relacionados con problemas de disipación de calor versus averías. La tendencia de los ordenadores transistorizados era disminuir en lo posible este consumo de potencia, que no comportaba ventajas de ningún tipo. Pese a todos estos inconvenientes que acabamos de indicar, el Gamma 60 era una máquina innovadora que supuso todo un hito en su época.

■ Otros procedimientos de control del proceso de inferencia



OS esquemas de búsqueda descritos representan procedimientos generales de gestión del proceso de inferencia en un sistema experto. Sin embargo, normalmente todo sistema concreto (o cualquier "herramienta"

de generación de sistemas) incluye algún procedimiento adicional de control. Vamos a ver algunos de estos sistemas.

Hay dos modos distintos de introducir un control adicional: mediante instrucciones y parámetros de control o a través de metarreglas. El primer modo consiste en añadir alguna instrucción que dirija el proceso de un modo concreto. El segundo se realiza introduciendo en la base de conocimiento ciertas reglas de producción cuyo procesamiento produce que se dirija la búsqueda por un camino o por otro.

Dentro del primer grupo podemos citar los siguientes mecanismos: restricciones en el uso de ciertas reglas, estrategias alternativas de búsqueda, introducción de hechos ficticios, órdenes de parada o pausa, etc. En cuanto a las "metarreglas", se puede hablar de reglas de "ver primero", de activación de paquetes de reglas, de manejo de contenidos, etc.

En algunos sistemas se puede introducir un parámetro en cualquier regla que indique al sistema que dicha regla *sólo puede ser utilizada en encadenamiento hacia adelante* (o

hacia atrás): entonces el motor de inferencia despreciará esa regla cuando esté realizando un encadenamiento hacia atrás (hacia adelante, respectivamente).

Otros sistemas disponen de varias *estrategias típicas que pueden ser elegidas alternativamente*. Por ejemplo: la estrategia consistente en que "la regla primera a elegir es aquella que tiene el menor número de condiciones todavía sin concretar", se llama MINCOND; "elegir la regla que tiene entre sus premisas el hecho que hayamos establecido más recientemente" es HECHREC; "tomar la regla que tiene más conclusiones" es MAXCONC, etcétera. De tal modo que podemos realizar dos tipos de control (según nos lo permita el generador que estemos utilizando o según lo programemos): detener el proceso en un momento dado e introducir la orden correspondiente desde el exterior del sistema o bien incluir en la base de conocimientos una regla que contenga esa información: "si da-leche y es un ungulado entonces HECHREC". Cuando se active esta regla, el motor de inferencia entiende la orden HECHREC y modifica la estrategia a aplicar a partir de entonces.

En ocasiones, se pueden introducir *hechos ficticios* para guiar el proceso de un modo diferente al previsto. Imaginemos que nuestro motor tiene previsto (como estrategia de búsqueda) seleccionar ante todo aquella regla, de entre las posibles, que tenga menos premisas (menos condiciones en el antecedente). Si disponemos de las reglas

- R1: Si vuela y
pone huevos → es un ave
- R2: Si tiene plumas → es un ave

el sistema seleccionará la segunda y después

TEMAS MONOGRAFICOS DE VANGUARDIA

la primera. Si deseamos, por alguna razón, que sean activadas en el orden en que aparecen, se puede modificar la base de conocimientos poniendo

- R1: Si vuela y
pone huevos → conclusión-1
- RF1: Si conclusión-1 → es un ave
- R2: Si tiene plumas → conclusión-2
- RF2: Si conclusión-2 → es un ave

el resultado será que al buscar en encadenamiento hacia atrás una regla que demuestre "es un ave", seleccionará ante todo la regla RF1 y para obtener su condición ("conclusión-1") realizará R1. Sólo después irá a examinar RF2 y R2. Hemos cambiado el orden previsto originalmente en el motor de inferencia.

Del mismo modo, hay sistemas que disponen de la *orden PARADA* o *PAUSA* que puede aparecer en cualquier regla de producción o como comando en el sistema.

Por otro lado, se pueden introducir en el sistema "*metarreglas*" o reglas con información sobre el proceso de las reglas. De este tipo son las *reglas de "ver primero"*; es decir, reglas del tipo "Si condición-1 y condición-2 ENTONCES ver-primero hecho-1 y hecho-2". Con esta regla se consigue cambiar el orden de búsqueda, dirigiendo el proceso hacia "hecho-1" y "hecho-2", en vez de seguir el proceso general previsto por el motor.

En ocasiones se introducen *reglas de activación de un paquete de reglas*. Si hay un grupo de reglas (un paquete) que sólo queremos que se active en unas condiciones determinadas, se puede incluir una condición fija en las premisas ("condición-de-paquete-1", por ejemplo), y cuando se dispare la regla dada (que comporta entre sus conclusiones la "condición-de-paquete-1") se pondrán en situación de ser activadas todas las reglas del paquete (todas las que tengan como premisa la tan repetida "condición-de-paquete-1"). Todo lo dicho sirve para encadenamiento hacia adelante. Si se está trabajando en encadenamiento hacia atrás, la regla de activación del paquete deberá tener la clave entre sus premisas y el paquete deberá tener en sus conclusiones dicha clave.

Por último, vamos a comentar que, en ocasiones, las metarreglas aluden no a la posición o tipo de las reglas de que se trata, sino que dirigen el funcionamiento del motor *aludiendo a las reglas por su contenido*. Por ejem-

plo, en el clásico MYCIN aparecen metarreglas como las siguientes

- R1. Si se busca una terapia,
ENTONCES considerar, por el orden dado, las reglas que permiten
 1. Adquirir las informaciones clínicas sobre el paciente;
 2. Descubrir qué organismos, si hay alguno, son la causa de la infección;
 3. Identificar los organismos más probables;
 4. Descubrir los medicamentos potencialmente útiles en el caso;
 5. Elegir los más adecuados y en menor número.
- R2. Si 1. El paciente es sujeto de alto riesgo;
 2. Existen reglas que mencionan las pseudománias en alguna de sus premisas;
 3. Existen reglas que mencionan las Klebsiellas en alguna de sus premisas,ENTONCES es probable que sea preferible utilizar antes las del tipo aludido en 2, que las aludidas en 3.

Hay dos elementos de conocimiento muy útiles, y representativos en numerosas ocasiones de las situaciones reales, a los que aún no hemos aludido todavía; queremos hacerlo ahora como conclusión de este apartado; son el proceso de la incertidumbre y el proceso no-monótono.

■ Razonamiento aproximado

Se alude con este nombre al proceso de los datos de incertidumbre que podemos (o debemos, en ocasiones) introducir en los sistemas.

En efecto, hay muchos casos en que al formalizar una aseveración (escribir una regla de producción) o bien al establecer un hecho, no se tiene la plena seguridad de que sea cierto lo que estamos afirmando. Esta circunstancia se suele incluir en la base de conocimiento mediante un factor de incertidumbre o de certeza. Puede aparecer a nivel del experto mediante la introducción del factor correspondiente en la regla o bien a nivel del usuario como un factor que afecta a los hechos que se definen como falsos o verdaderos.

Un ejemplo del primer caso sería la regla

"si el cultivo es sangre y la forma del microorganismo es bastón entonces es probable (0,6) que el organismo sea un pseudomonas aeruginosa"

Ya hemos aludido a ello anteriormente, así como a las escalas (de -1 a +1 o de -100 a +100) que se suelen utilizar. Aquí, comentando la estructura del motor de inferencia, hemos de aludir a cómo se manejan estos datos en el proceso deductivo.

Existen numerosos procedimientos de gestionar los factores de certeza según la finalidad del sistema experto de que se trate, pero los elementos que se consideran, básicamente, son dos: umbral de activación de una regla y "propagación" de la incertidumbre en los procesos deductivos.

Cuando no se realiza razonamiento impreciso o aproximado, una regla se activa si son ciertas todas las premisas. Cuando los hechos se conocen, por el contrario, afectados de un factor de fiabilidad ("es cierto que el animal correspondiente da leche —coeficiente 0,7—") hay que establecer un límite inferior (umbral) a partir del cual se considera que los hechos son "suficientemente ciertos" como para activar la regla correspondiente. Es usual que haya disponibles instrucciones para modificar ese umbral bien de un modo general o a través de metarreglas.

Por otro lado, el motor de inferencia debe tener establecidos unos mecanismos de cálculo de los coeficientes de fiabilidad para los hechos deducidos en función de los coeficientes de las premisas y/o del coeficiente general de la regla de producción correspondiente. El mecanismo más utilizado en la mayoría de los sistemas es el de "propagación multiplicativa con acumulación de plausibilidad (o certeza)". Ya en MYCIN se utilizó este procedimiento, que ha aparecido posteriormente en numerosos sistemas, con un esquema semejante al siguiente:

a) Combinación de certezas para una sola regla que concluye de un hecho:

a.1.) si la conclusión no tiene coeficiente;

hecho-1 y hecho-2 → hecho-3

(cf = 0,5) (cf = 0,3) (cf = 0,3)

coeficiente de la conclusión igual al mínimo de los coeficientes de las premisas.

a.2.) si la conclusión tiene coeficiente de certeza;

hecho-1 y hecho-2 → hecho-3 (con factor 0,5)

(cf = 0,5) (cf = 0,3) (cf = 0,15)

coeficiente de la conclusión igual al mínimo de los coeficientes de las premisas (0,3) multiplicado por el coeficiente de la regla (0,5). (Esta es la propagación multiplicativa).

b) Acumulación de plausibilidad cuando varias reglas concluyen sobre un mismo hecho:

Si la regla R1 concluye hecho-3 con coeficiente cf=0,3 y la regla R2 concluye hecho-3 con coeficiente cf=0,2

se produce una "acumulación" de la certeza sobre este hecho-3 ("va apareciendo" progresivamente como más cierto cada vez). El factor acumulado que se atribuye al hecho-3 será de 0,44 según la siguiente regla:

$$cf_c = cf_1 + cf_2 - cf_1 * cf_2$$

(si cf_1 y cf_2 son positivos)

(donde cf_1 es el coeficiente de certeza de C cuando se obtiene como conclusión de la R1; cf_2 el correspondiente valor de C obtenido a partir de la R2 y cf_c el coeficiente acumulado que se calcula)

$$cf_c = cf_1 + cf_2 + cf_1 * cf_2$$

(si son ambos negativos)

$$y \text{ } cf_c = cf_1 + cf_2 / (1 - \min(|cf_1|, |cf_2|))$$

(si son de diferente signo)

(siendo $\min(|cf_1|, |cf_2|)$ el menor de los valores absolutos de cf_1 y cf_2)

Esta regla se puede justificar de un modo gráfico diciendo que si tenemos una certeza de 0,3 para el hecho C (a partir de la regla 1) y se nos ofrece una plausibilidad adicional (de la regla R2) de 0,2, se incrementa la certeza que teníamos "aproximándola" hacia la certeza absoluta (valor cf=1) el 20% de la cantidad que le "faltaba" para llegar a 1 (en nuestro caso "faltaba" 0,7 y, por tanto, se produce un incremento de certeza de 0,14.

■ Procesos no monótonos

Los procesos deductivos que hemos considerado hasta ahora se han realizado aceptando que en cada "sesión de trabajo" los hechos eran verdaderos o falsos de un modo estable, de manera que si llegábamos a concluir su valor, se fijaban en ese valor y no se volvían a modificar (incluso se "tachaban" todas las reglas correspondientes). Este tipo de razonamiento se llama "monótono", pues de hecho, la base de conocimientos crece de un

modo permanente ("monótono") y nunca decrece.

Sin embargo, no repugna la idea de obtener provisionalmente que "el animal es un ungulado" (con alguna finalidad demostrativa concreta) para llegar a la conclusión, posteriormente, de que no lo es. Parece que se adecúa más a la realidad de la actividad deductiva esta flexibilidad: es decir, preferiríamos que nuestro motor de inferencia fuera no-monótono. (En los procesos de establecimientos

de planes, por ejemplo, esta capacidad de elaborar hipótesis que luego son evaluadas y, eventualmente, rechazadas es básica).

De hecho, ya existen sistemas que aceptan esta posibilidad (con algunas restricciones) y sería deseable poder trabajar siempre en esta línea; pero las dificultades de manejo del conocimiento en los sistemas no-monótonos es grande y la mayoría de los sistemas trabajan de modo monótono exclusivamente.

TERMINOLOGIA

Arbol de contexto. También llamado árbol objeto. En EMYCIN, el árbol de contexto constituye la espina dorsal del programa de consulta. Consiste en una organización estructurada de los objetos (contextos) o entidades conceptuales que constituyen el dominio de consulta. Pueden existir uno o más contextos. El árbol de contextos estático es una organización de tipos de contextos (por ejemplo, un paciente al que se le han preparado cultivos). Un árbol de contexto dinámico es un conjunto de instancias o especificaciones de contextos (por ejemplo, Juan Pérez con un cultivo realizado por la mañana y otro cultivo realizado por la tarde).

Arbol de objetivos. Estructura de datos en forma de árbol, en la que el nodo raíz representa un objetivo que debe alcanzarse, y las ramas hijas de cada objetivo representan subobjetivos, que cuando se alcancen serán suficientes para alcanzar el objetivo representado por su padre. Los árboles de objetivos pueden ser árboles and/or.

Cadena de inferencia. Secuencia de pasos o reglas de aplicación utilizados por un sistema basado en reglas para llegar a conclusiones.

Concordancia. En un sistema de producción, el proceso de concordancia compara un conjunto de patrones que aparecen en el lado izquierdo de las reglas, con los datos de la memoria de datos, para ver todos los caminos posibles en los que se pueden satisfacer las reglas con ligaduras consistentes.

Concordancia parcial. Conjunto de asociaciones entre los elementos de condición (antecedentes de las reglas) y los elementos de la memoria de trabajo, que satisfacen parcialmente el lado izquierdo de la regla. No todos los elementos de condición necesitan ser contrastados. En algunos casos se da un umbral para especificar el número mínimo que debe ser contrastado.

Condición. a) Antecedentes de una regla; b) Elemento de una condición; c) Proposición que resume el estado de ejecución de un programa.

Dominio. En matemáticas, conjunto de valores que debe asumir el argumento de una función. En sistemas expertos, el campo del conocimiento o clase de las tareas que puede abordar.

Elemento de la memoria de trabajo. Es la unidad de la memoria de trabajo. En los lenguajes orientados al objeto, los elementos de la memoria de trabajo son elementos de valor-atributo.

Esquema. Formalismo para representar una información sobre un concepto único, utilizando propiedades relacionadas con dicho concepto. Las propiedades se suelen representar mediante una ranura, y pueden consistir en procedimientos relacionados con ellas para computar las propiedades que no están disponibles inmediatamente.

Ingeniería del conocimiento. Proceso de construcción de sistemas expertos.

TERMINOLOGIA

Lenguaje de ingeniería del conocimiento de propósito general. Lenguaje de ordenador diseñado para construir sistemas expertos e incorporarles características que los hacen aplicables a distintos problemas, áreas y tipos.

Lenguaje orientado al problema. Lenguaje de ordenador diseñado para una clase de problemas específicos, por ejemplo, el FORTRAN, diseñado especialmente para realizar cálculos científicos (específicamente, algebraicos), y el COBOL, con características muy útiles para el desarrollo de aplicaciones empresariales.

Metarregla. Regla relacionada con los conocimientos de metanivel. Las metarreglas se utilizan para especificar las estrategias de resolución de conflictos, o para filtrar y ordenar reglas del dominio.

Métodos orientados al procedimiento. Métodos de programación que utilizan subrutinas anidadas para organizar y controlar la ejecución del programa.

Módulo de explicaciones. Parte de un sistema experto que explica cómo se alcanzan

las soluciones y justifica los pasos utilizados para alcanzarlas.

Ranura. Componente de un objeto en un sistema de marcos. Las ranuras pueden contener características intrínsecas como, por ejemplo, el nombre del objeto, atributos y valores, atributos con valores por omisión, indicadores que aluden a otras estructuras relacionadas e información sobre el creador de la estructura, etc.

Razonamiento no monotónico. Técnica de razonamiento que soporta múltiples líneas de razonamiento (varios caminos para alcanzar la misma conclusión) y en la que se admite la posibilidad de asignar nuevos valores a hechos o conclusiones aparte de una nueva información obtenida. Es útil para procesar conocimientos y datos que no ofrecen confianza.

Redundancia temporal. Tendencia de los sistemas de producción a realizar pocos cambios en la memoria de datos (y, por tanto, en el conjunto conflicto) en el espacio de tiempo que va desde un ciclo de reconocimiento al siguiente.

